MICROCOMPUTADORES (y II)

NUEVAS TECNOLOGIAS

BIBLIOTECA DE ELECTRONICA/INFORMATICA



marcombo



BIBLIOTECA DE ELECTRONICA/INFORMATICA

MICROCOMPUTADORES
(y III)



Esta obra es una nueva edición actualizada y ampliada de la obra originalmente publicada por Marcombo, S.A. de Boixareu editores, con el título de «Aplicaciones de la Electrónica»

El contenido de la presente obra ha sido realizado por Marcombo, S.A. de Boixareu editores, bajo la dirección técnica de José Mompin Poblet, director de la revista «Mundo Electrónico»

ISBN 84-7634-485-6 (Obra completa)
ISBN 84-7634-735-9 (Vol. 37)
D.L.: B. 28761-1986

Impreso y encuadernado por
printer industria gráfica, sa c.n. II, cuatro caminos, s/n
08620 sant vicenç dels horts barcelona 1986

Printed in Spain

Microcomputadores (y II)

ACCESO DIRECTO A MEMORIA

Todos los métodos de control de transferencia de información entre microprocesadores (como unidad central de un microcomputador) y sus periféricos estudiados hasta ahora eran llevados a cabo por el microprocesador, de forma que siempre era éste quien ejecutaba la instrucción de transferencia correspondiente.



Los minicomputadores han desbancado, en muchas aplicaciones, a los grandes computadores. Estos últimos quedan casi olvidados en lo que respecta al gran público y a la pequeña y mediana empresa.

Si bien esta situación no es limitativa en el caso de que la información que se deba intercambiar no sea excesiva, cuando el periférico deba intercambiar una gran cantidad de información con la memoria la pérdida de efectividad que proporcionan los métodos anteriores es evidente.

Para permitir al periférico que dialogue de forma directa con la memoria se ha desarrollado una técnica especial denominada de *acceso directo a memoria* o DMA *(Direct Memory Access)*

Esta estructura complica bastante el interface de acoplamiento entre periférico y unidad central, ya que dicha unidad de acoplamiento debe simular en parte las funciones de la unidad de control de la unidad central cuando el periférico accede directamente a memoria sin contar con el microprocesador. Todos los conceptos tratados en el estudio de las interrupciones son válidos en el caso de acceso directo a memoria para varios periféricos, si bien es cierto que también se presentan elementos diferenciales de interés.

Estudiemos con detalle cada uno de estos aspectos.

La situación más sencilla se produce cuando determinado periférico intercambia información con una zona concreta e invariable de la memoria. Este problema no presenta ningún inconveniente especial y es tratado de modo clásico atendiendo a las técnicas de acceso directo a memoria.

Otro paso más adelante nos lleva a considerar el caso, más frecuente por otro lado que el anterior, de que sea variable la zona de memoria con la que intercambia información el periférico. En este caso es la unidad central la que decide iniciar el acceso directo a memoria, indicando a través de su unidad de control cuál es la posición en la que deben iniciarse las transferencias con la memoria, incluyendo a veces la posición final.

El control del acceso directo a memoria, en cualquiera de las dos posibilidades anteriores, puede llevarse a cabo de diversas formas, lo cual condiciona como se verá la operatividad de la unidad central.

El control de acceso directo a memoria se puede llevar a efecto por tiempo completo, en cuyo caso durante todo el tiempo en que el periférico está intercambiando información con el periférico, no puede hacerlo el microprocesador (unidad central).

Otro procedimiento consiste en aprovechar los ciclos de ejecución de una instrucción durante los cuales el microprocesador no accede a datos de la memoria. Durante estos ciclos puede hacerse que el periférico acceda a la memoria combinándose el acceso a la memoria entre el microprocesador y el periférico.

Las técnicas de acceso directo a memoria son también

muy estudiadas ya que de ellas depende, como el lector habrá adivinado, buena parte de la optimización operativa de los microcomputadores. En la actualidad, muchos son ya los sistemas que incorporan esta característica de DMA, lo cual permite unas prestaciones de los microcomputadores mucho mejores de las que no la incorporan.

SISTEMAS PERIFERICOS PROGRAMABLES

Todos los bloques funcionales que se han tratado hasta ahora son de validez general, todos ellos realizables y existentes en circuitos de media y larga escala de integración (MSI y LSI).



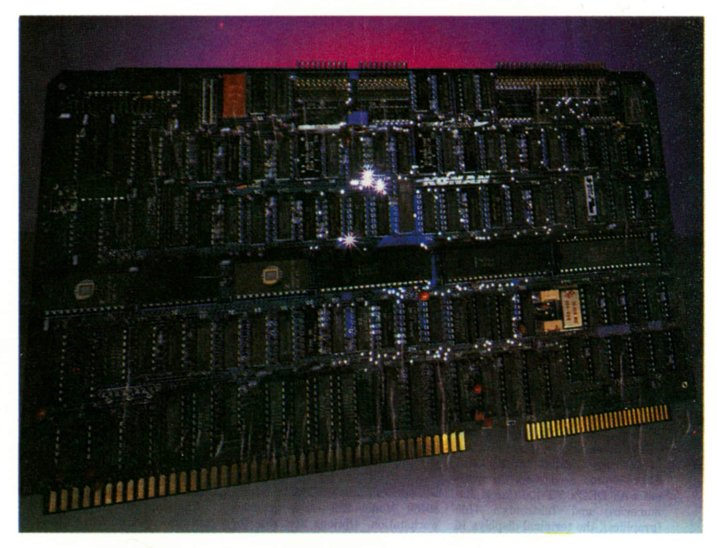
La aplicación de los microcomputadores como ayuda al diseño es muy importante en la actualidad y las expectativas de futuro son aún mejores todavía. En la fotografía puede verse un sistema CAD (diseño asistido por computador).

Muchos de estos bloques, cuyos aspectos genéricos se han tratado, son programables según se ha visto, habida cuenta de las funciones que deben realizar. Estas, en lo que a la transferencia de información entre periféricos y unidad central se refieren pueden ser de varios tipos:

— Funciones de transferencia, cuyos dispositivos encargados de ejecutarla estarán especializados en tales menesteres de transferencia de información, ya sea en los modos serie o paralelo o bien como transmisión o recepción de información.

— Funciones de control de la transferencia, cuyos dispositivos encargados de efectuar dicho control estarán especializados en llevarlo a cabo en la forma adecuada. También se han efectuado en circuitos integrados de gran escala de integración.

Finalmente existen también dispositivos especializados en el control y en la ejecución de transferencia de información.



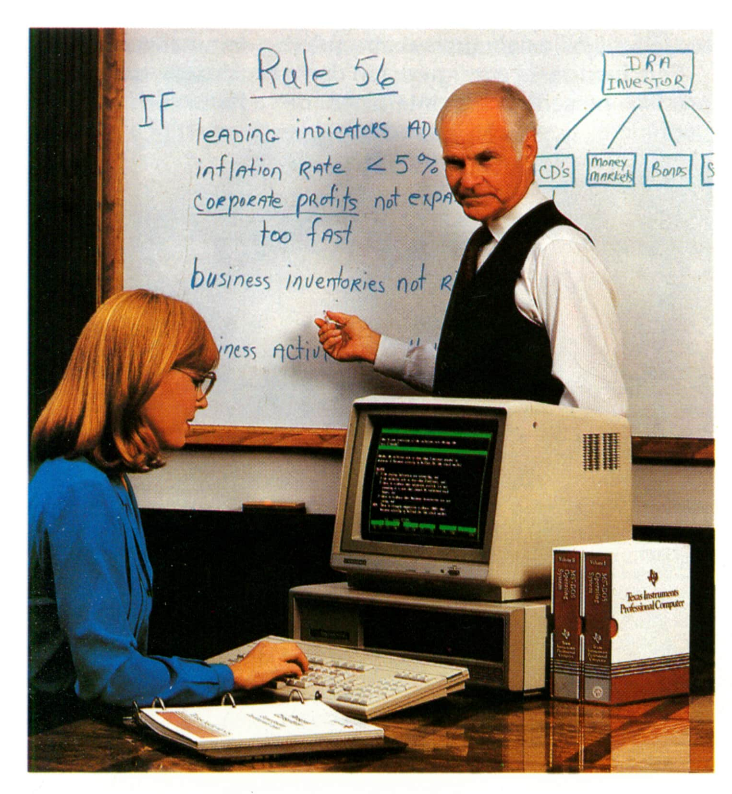
En una placa de interface a un periférico pueden contemplarse infinidad de circuitos integrados LSI (algunos de ellos programables). En la fotografía se observa un controlador de disco duro de 5 1/4 pulgadas, desarrollado por la firma Konan.

Es destacable que la mayoría de estos sistemas son programables con el fin de adaptar su funcionalidad a distintas situaciones, sin deber por ello modificar su estructura física o hardware.

Los circuitos periféricos programables son, como el lector ha podido comprobar, dispositivos imprescindibles para el buen funcionamiento de los microcomputadores y de la evolución de dichos sistemas depende en buena parte la futura evolución de la *microinformática*.

A modo de recopilación convendrá recordar que en este capítulo se han estudiado, desde un punto de vista funcional y estructural, la problemática de transferencia de información entre la unidad central y la memoria de un sistema programable, así còmo la transferencia de información entre

unidad central y periféricos y entre éstos y la memoria, siendo todos ellos elementos básicos de que está constituido un microcomputador.



Los computadores para aplicaciones típicas de oficina son los más generalizados. El computador profesional de Texas Instruments es un claro ejemplo de lo expuesto.

¿COMO SE CONFIGURA UN MICROCOMPUTADOR?

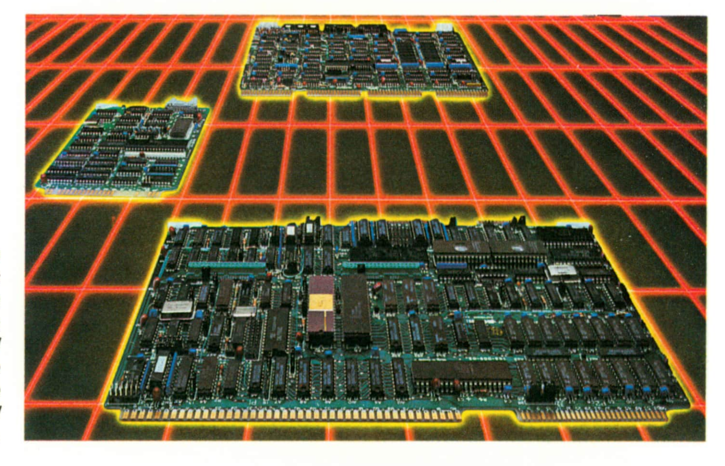
Cuando se pretende desarrollar prácticamente un microcomputador es el momento de evaluar los aspectos técnicos y funcionales que permiten especificarlo.

Un microcomputador se configura a partir de un hardware, que constituye la parte electrónica del mismo, y de un software o programas de ayuda (como es el sistema operativo, manipuladores de periféricos, etc.).

Dada esta configuración, el usuario o analista de aplicaciones deberá desarrollar los programas de trabajo o programas de aplicación que se precisen.

De esta forma el microcomputador puede definirse como un sistema pensante del que únicamente es preciso conocer los lenguajes de programación que puede entender.

Para finalizar la configuración de este microcomputador es preciso prescribir los interfaces de adaptación con los periféricos, que serán los responsables de permitir el diálogo entre la máquina (computador) y el hombre. Tras la exposición de todos estos conceptos preliminares, vamos a entrar a continuación en un análisis más detallado de la configuración de un microcomputador.



La estructura hardware de los microcomputadores está condicionada fundamentalmente por la elección del microprocesador que realizará las funciones de Unidad Central del sistema (CPU).

En primer lugar es básica la elección del microprocesador que va a usarse como unidad central. Esta elección será función, principalmente, del sector de aplicación al que se pretenda destinar de forma prioritaria el microcomputador; no obstante, las consideraciones fundamentales a tener en cuenta en la elección del microprocesador desde la óptica del microcomputador que pretende configurarse son las siguientes:

— La velocidad de trabajo del sistema será función directa de la velocidad en la ejecución de las instrucciones por parte del microprocesador.

— La memoria con que vaya a dotarse al sistema, en las dos opciones de memoria activa y pasiva. Cada microprocesador tiene diferentes modos y capacidad de direccionamiento y ello condiciona grandemente su elección.
— Capacidad y niveles de interrupción, en funcionamiento con los distintos periféricos a los que deba interconectarse el microcomputador.



Computador destinado a manejar una gran cantidad de datos. Este terminal suele estar en conexión con otro computador central. (Cortesía: Facom).

— Ayudas al diseño que pueden disponerse para el desarrollo de sistemas en los que se aplica el microprocesador. Los sistemas de desarrollo son básicos en este caso y su importancia operativa para la simulación de funciones es grande.
— Ayudas a la programación, considerándose esta cualidad como trascendental para el posterior uso del microcomputador.

— Estructura de las entradas y salidas, aspecto que condiciona grandemente la posterior aplicación de la configuración obtenida.

Es claro que la elección del microprocesador no es la única consideración a tener en cuenta para desarrollar un microcomputador, si bien es necesario señalar que es el aspecto más importante a tener en cuenta, ya que todos los aspectos del hardware y del software girarán en torno al microprocesador seleccionado.

A partir de éste, la interconexión de todos los circuitos, así como el acceso del microcomputador al mundo exterior a través de las entradas y salidas, configurarán la estructura hardware del microcomputador. El sistema operativo y el software de ayuda constituirán la estructura software del mismo. Estos dos elementos, hardware y software forman la estructura operativa de un microcomputador.

El resto de este libro va a ocuparse de abordar el estudio práctico de la obtención de una estructura del hardware de un microcomputador y tratará los aspectos del software relativos al mismo. Es de señalar que algunos de los conceptos que se tratan a continuación han sido ya abordados en capítulos anteriores, pero ahora lo serán desde la óptica de un computador completo dispuesto a efectuar determinadas funciones.

REALIZACION PRACTICA DE INTERCONEXION ENTRE LA UNIDAD CENTRAL Y LAS MEMORIAS E INTERFACES

Un microcomputador no debe ser únicamente un sistema capaz de ejecutar determinadas instrucciones o llevar a cabo ciertas tareas. Es imprescindible que sea capaz de intercomunicarse con el mundo exterior y «entender» de él aquella información que le sea precisa y «hacerse entender» también de forma conveniente.

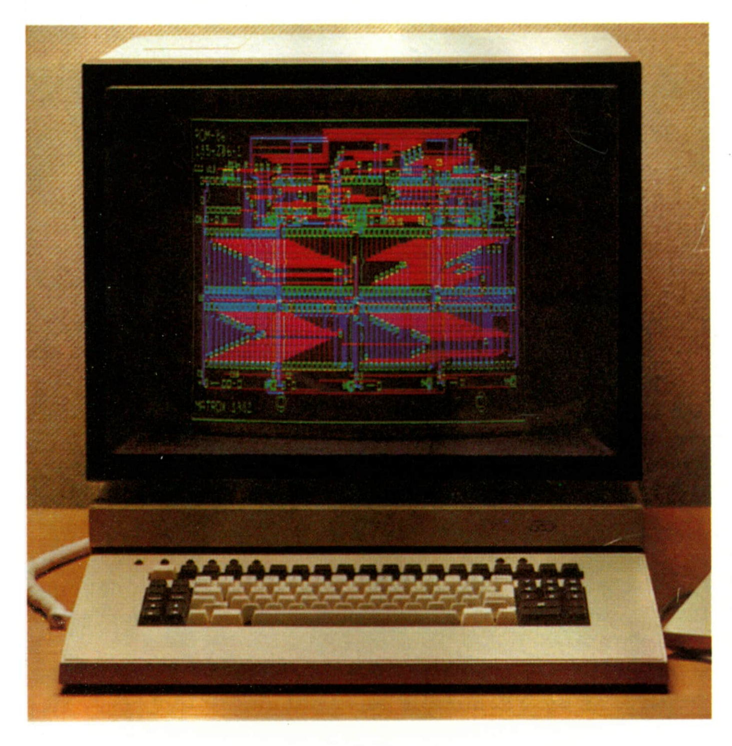
En torno a estas dos ideas fundamentales gira toda la problemática de los microcomputadores.

En el libro anterior se trató a fondo lo relativo a la interconexión entre la unidad central, la memoria y los periféricos, pero todo ello se hizo desde un prisma generalizador y fundamental. En este apartado va a tratarse este tema desde una óptica concreta, tratando no obstante de no

perder de vista aquellos aspectos de interés genérico o de aplicación común a todos los microcomputadores.

Un microcomputador no puede funcionar sin que exista una intercomunicación entre las diferentes partes que lo constituyen. Obviamente esta intercomunicación ha de ser rápida, ya que en caso contrario se perdería parte de la operatividad de estos sistemas.

Con el fin de permitir esta intercomunicación existen los buses, que son los soportes físicos de las señales eléctricas que intercambian entre sí los circuitos de los microcomputadores.



Existen sistemas operativos especializados en aplicaciones concretas. En la fotografía se muestra un microcomputador utilizado como sistema CAD.

Según la información que se presente por los buses, la unidad de control especifica qué información y entre qué dispositivos se debe intercambiar la referida información. La configuración resultante de los circuitos y buses de

interconexión es lo que constituye (dejando aparte dispositivos de alimentación u otros accesorios) la estructura de un microcomputador. Aunque la estructura de cada microcomputador es según se ha dicho distinta, existen tres funciones básicas que están presentes y deben ser realizadas por todo microcomputador:

— Enviar direcciones de la unidad central a las memorias y periféricos.
— Entrar datos a la unidad central procedentes de la memoria y periféricos.
— Sacar datos de la unidad central procedentes de la memoria y los periféricos.

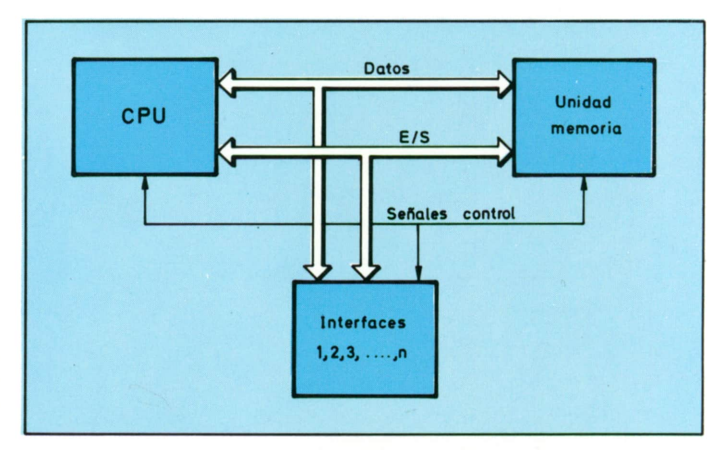


Figura 8. Estructura básica de un microcomputador con buses independientes.

Es claro, y es conveniente insistir en ello, que cada microcomputador puede ofrecer estructuras distintas a la presentada o incluso más complejas que la que se muestra en la figura 9.

Ya en el libro anterior se profundizó sobre las diferentes estructuras de bus único y separado, por lo que no se va a insistir más en aspectos estructurales, abordando en este libro otros puntos de detalle con ejemplos concretos.

Se vio en el tema anterior la problemática que aparecía en el caso de que los periféricos estuvieran conectados a un bus común. Para que el circuito de interface pueda reconocer si la orden está dirigida a él o no, se puede realizar de una

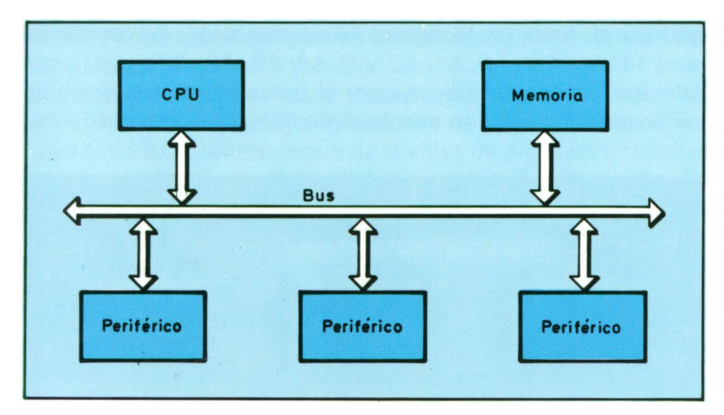


Figura 9. Estructura básica de un microcomputador con un bus único.

forma sencilla un circuito discriminador que ponga en línea al periférico únicamente cuando se dirige la información a él (figura 10).

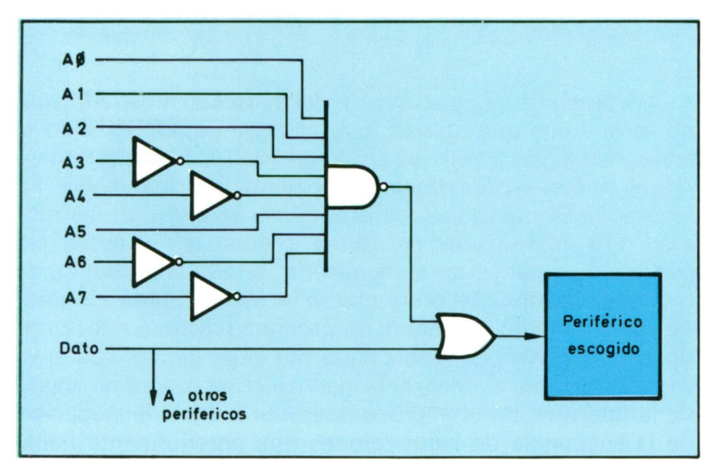
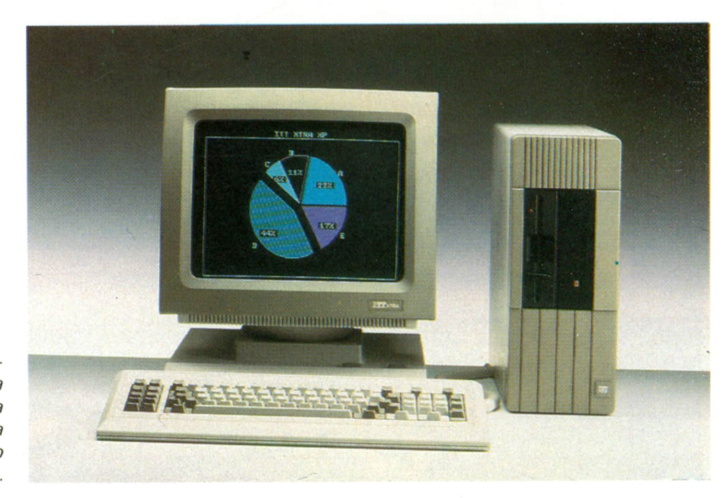


Figura 10. Circuito de identificación de periférico. En este caso identifica el periférico 11100100.

El discriminador de direcciones de periféricos propuesto consta de una puerta NAND y de un determinado número de inversores de forma que, según sea la dirección concreta del periférico, la salida de la puerta NAND será «0» cuando la dirección del periférico se corresponda con la del dispositi-

vo. En el caso de la figura 10 la dirección del periférico será 11100100 (A0, A1, A2, A3, A4; A5, A6, A7), por lo que la información que transite por el bus correspondiente sólo alcanzará el periférico con la dirección seleccionada.



Microcomputador ITT-XTRA-XP, dotado con la unidad periférica para recoger o procesar la información en diskettes o discos flexibles.

El proceso de reconocimiento del periférico mostrado aquí no es el único que utilizan los microcomputadores, pero el lector habrá apreciado que su simplicidad es notable, motivo por el cual este procedimiento se utiliza con frecuencia. En los microcomputadores los programas se ejecutan instrucción tras instrucción de forma secuencial. Cuando un periférico reclama la atención de la unidad central, la secuencia normal del programa se ve interrumpida y a partir de este momento se ejecuta un programa distinto relacionado con la interrupción solicitada por el periférico. Como ya se ha explicado, al finalizar la ejecución del programa objeto de la interrupción el microcomputador reinicia la ejecución de la secuencia de instrucciones que anteriormente había abandonado.

Las razones por las que existen los sistemas de interrupciones en los microcomputadores son de diversa índole, pero se pueden sintetizar en una más rápida respuesta del sistema a fenómenos externos al mismo y, en general, a un mejor aprovechamiento de las posibilidades de los microcomputadores.

Estos conceptos, aunque es fundamental tienen una clara consolidación en los siguientes datos:

Los microcomputadores, como sistemas con una alta velocidad de operación, son capaces de ejecutar instrucciones elementales en períodos de tiempo de algunos microsegundos. Habida cuenta que los tiempos de respuesta de periféricos clásicos como teletipos, relés, discos, etc., son de algunos milisegundos, es obvio que mientras el periférico responde a una orden del microcomputador, éste puede estar aprovechando el tiempo para ejecutar otras instrucciones y realizar otras funciones.

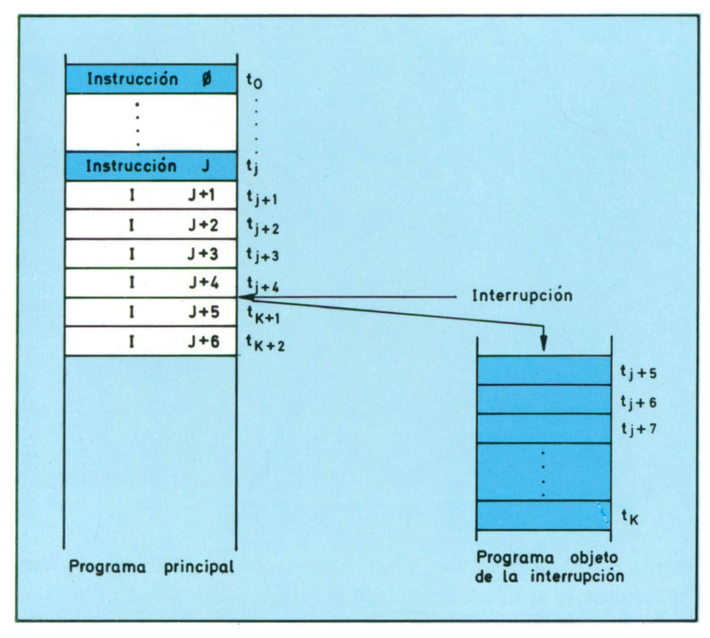


Figura 12. Ejecución de un programa de interrupción, con ruptura de secuencia del programa principal, y retorno a la instrucción siguiente.

No sólo la faceta de mejor aprovechamiento de la capacidad de los computadores es causa de la importancia de un adecuado sistema de interrupciones en un microcomputador, sino que es también de gran importancia la idoneidad del microcomputador para responder a influencias externas que, en casos de necesidad, precisen una

acción inmediata del mismo. Ejemplos de una situación de este tipo pueden darse por cualquier alarma en el control de un proceso industrial, en una necesidad prioritaria de detener el proceso a través de un periférico, etc. Como el lector puede imaginar el número de sistemas de interrupción es muy grande.

En los casos en que los periféricos del microcomputador tienen velocidades muy diversas, suele utilizarse un procedimiento mediante el que se agrupan las interrupciones en diferentes niveles de prioridad. Mediante la utilización de registros auxiliares de señalización (los *flags* o *banderas*) se pueden jerarquizar distintos sistemas de interrupciones, de



Aplicación de los microcomputadores destinados a intercomunicadores. La telemática permite, entre otras cosas, transmitir datos almacenados en un microcomputador por vía telefónica. (Cortesía: Standard Eléctrica).

forma que sea posible excluir interrupciones de menor nivel hasta que la que corresponda por su grado sea atendida.

Está claro que un sistema por identificación de las interrupciones es un procedimiento sencillo pero lento, tanto más en aquellos casos en que el número de interrupciones es grande. Es por ello que en la ejecución práctica de los sistemas de interrupción de los microcomputadores, se trata de aumentar la velocidad de respuesta mediante la grabación en memorias ROM (no volátiles) de instrucciones especiales de entrada/salida.

Existen otros procedimientos, como el de la identificación

vectorizada que, aunque menos usado, ofrece velocidades de respuesta apreciables.

Los sistemas de interrupción optimizan el funcionamiento de los microcomputadores, pero su realización complica grandemente tanto el hardware como el software del mismo. Por este motivo, antes de decidir aleatoriamente la realización práctica de un sistema de interrupciones en un microcomputador debe tenerse muy en cuenta si ello es



Modem de transmisión de datos de Standard Eléctrica. Permite la utilización de la línea mediante comunicación telefónica empleando varios canales.

realmente necesario, pues el sobrecoste, tanto del sistema como de su puesta en marcha, debe justificar su utilización real posterior. Es conveniente señalar también que los sistemas de interrupción son tanto más importantes cuanto

más potencia tenga el microcomputador por razones obvias de aprovechamiento. Puede llegar el caso en que sea más conveniente la utilización de diversos microcomputadores para distintas aplicaciones que precisen interrupciones distintas, capaces de integrar en un solo computador todo un sistema de interrupciones relativamente complejo.

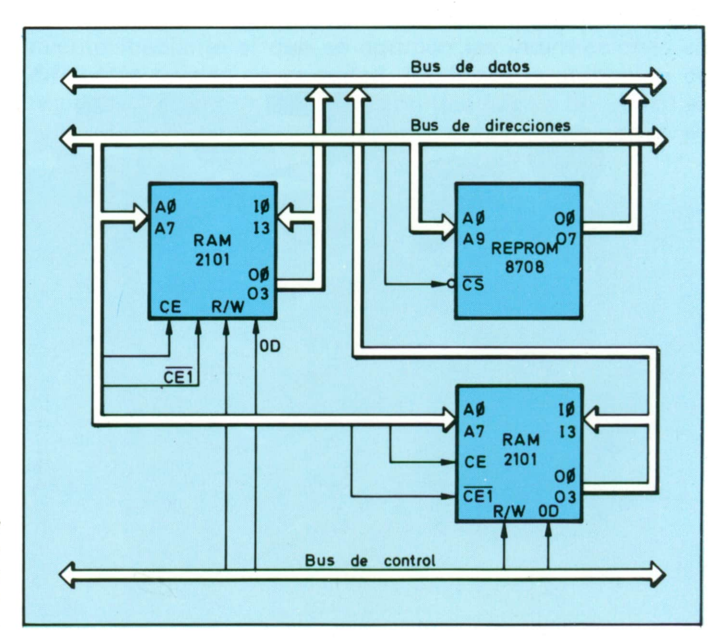


Figura 15. Conexión real de memorias RAM y REPROM al sistema de buses de un microcomputador.

Se ha hablado ya abundantemente a lo largo de éste y anteriores libros de la problemática de acoplamiento de la memoria y los periféricos a la unidad central de un microcomputador. Pasemos a continuación a un terreno concreto.

En la figura 15 puede apreciarse la conexión de memorias RAM estáticas (tipo 2101 o equivalente) y REPROM al sistema de buses de un microcomputador. Habida cuenta que la conexión de la unidad central a las memorias es una configuración interna a la estructura del propio microcomputador, se considerará a éste como un bloque que, interconectado al mundo exterior, intercambia

información con él. Es obvio que esta interconexión con el mundo exterior se llevará a cabo a través de unidades de entrada/salida que actúan como interfaces.

Analizando esta problemática sobre la estructura del microcomputador didáctico SDK-85 de Intel, podrá el lector profundizar en la ejecución práctica de interconexión de un microcomputador.

En la figura 16 puede apreciarse el diagrama de bloques del sistema SDK-85. Naturalmente que en las pró ximas líneas no se repetirá la descripción de la CPU 8085 que ya fue tratada anteriormente en el libro Microprocesadores I, por lo que va a pasarse a la descripción funcional del resto de dispositivos periféricos.

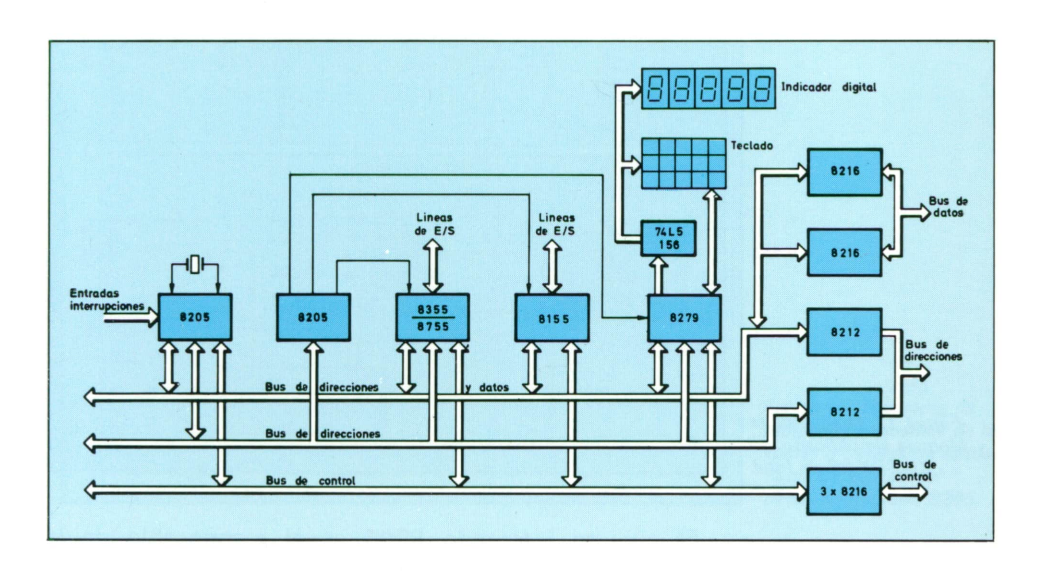


Figura 16. Diagrama de bloques del kit microcomputador SDK 85. Este kit está prescrito especialmente para la enseñanza.

El circuito integrado 8155 es un dispositivo LSI que está especialmente diseñado para ser compatible directamente con la CPU 8085. Puede almacenar 256 bytes (RAM estática) y dispone de 22 líneas de entrada/salida programables y de un contador de 14 bits. Esta memoria se usa para guardar en su interior programas del usuario o bien información que interese almacenar temporalmente. El contador interno que incorpora el 8155 se usa en la rutina

llamada «paso a paso» del monitor del SDK-85, con el fin de interrumpir a la CPU después de la ejecución de cada una de las instrucciones.

El circuito integrado 8755 es una memoria EPROM reprogramable, cuyo contenido puede borrarse mediante radiación ultravioleta. Puede almacenar 2048 bytes (2 K) y dispone de 16 líneas de entrada/salida. Como se aprecia por su diagrama de bloques, el SDK-85 lleva un 8755 con espacio para ubicar un segundo a voluntad del usuario.

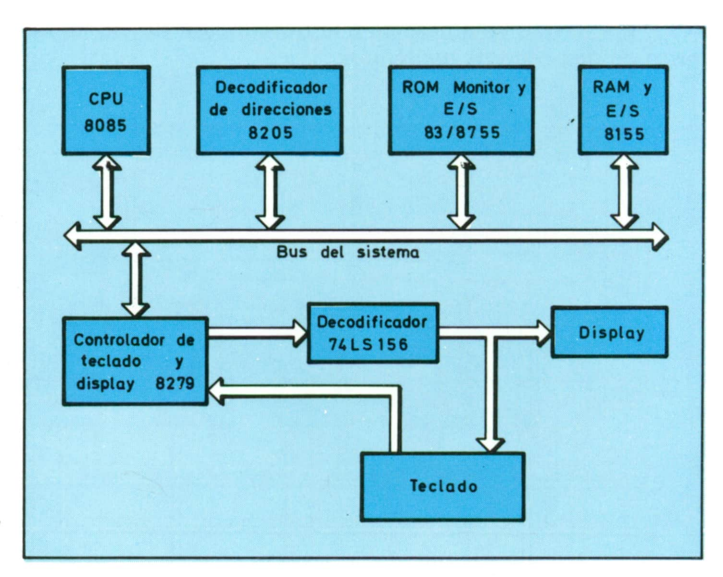
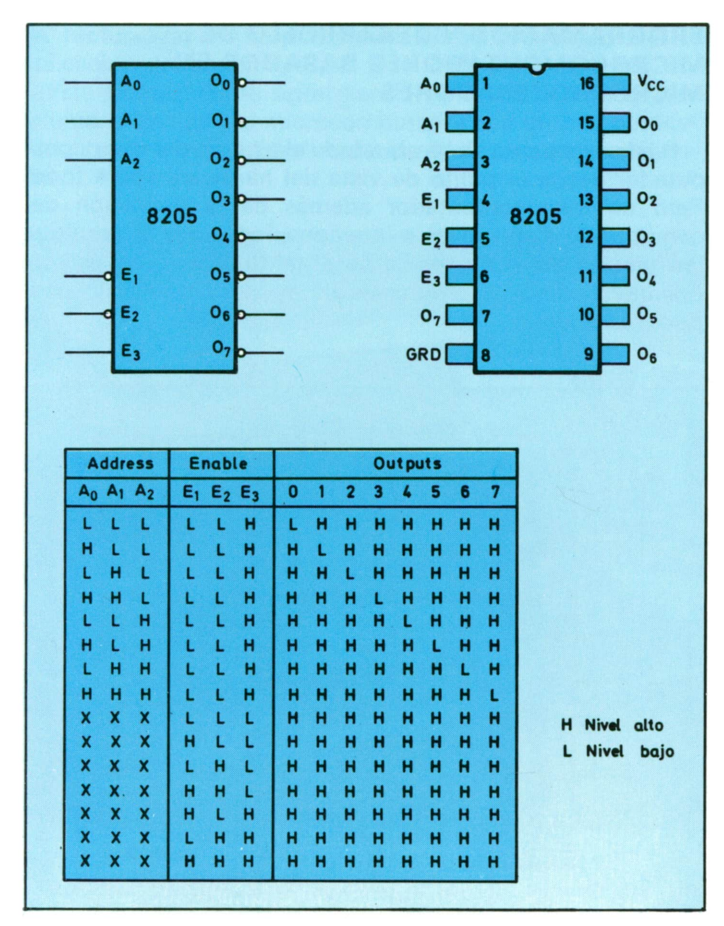


Figura 17. Diagrama de bloques simplificado del Kit SDK 85 de Intel.

El circuito integrado 8205 es el responsable de la decodificación de los bits de direccionamiento de memoria para activación de los circuitos 8155, 8755 y 8279. Este último dispositivo es el que se encarga del control del teclado y bloque de visualización, de forma que actúa de interface entre la CPU 8085 y la unidad de visualización y teclado del SDK-85. El 8279 es el responsable de mantener por refresco la unidad de visualización a través del contenido de una unidad de memoria interna, mientras que simultáneamente muestrea el teclado con el fin de detectar cualquier posible entrada.



Address			Enable			Outputs							
A_0	A_1	A_2	E_1	E_2	E_3	0	1	2	3	4	5	6	7
L	L	L	L	L	H	L	H	H	H	H	H	H	H
H	L	L	L	L	H	H	L	H	H	H	H	H	H
L	H	L	L	L	H	H	H	L	H	H	H	H	H
H	H	L	L	L	H	H	H	H	L	H	H	H	H
L	L	H	L	L	H	H	H	H	H	L	H	H	H
H	L	H	L	L	H	H	H	H	H	H	L	H	H
L	H	H	L	L	H	H	H	H	H	H	H	L	H
H	H	H	L	L	H	H	H	H	H	H	H	H	L
X	X	X	L	L	L	H	H	H	H	H	H	H	H
X	X	X	H	L	L	H	H	H	H	H	H	H	H
X	X	X	L	H	L	H	H	H	H	H	H	H	H
X	X	X	H	H	L	H	H	H	H	H	H	H	H
X	X	X	H	L	H	H	H	H	H	H	H	H	H
X	X	X	L	H	H	H	H	H	H	H	H	H	H
X	X	X	H	H	H	H	H	H	H	H	H	H	H

Figura 18. Diagrama de bloques y esquemas de los «pins» o patillas de salida. Tabla de verdad del circuito integrado 8205.

Con la descripción sintetizada del microcomputador didáctico SDK-85, el lector habrá constatado que la complejidad de los microcomputadores es notable, aun en el caso en que los periféricos se reducen a una unidad de visualización y un teclado, y la problemática de interconexión resulta bastante simple.

La estructura de los microcomputadores profesionales es mucho más compleja que la reseñada, pero conserva una arquitectura básica equivalente a la descrita en aquel caso, siempre que equipen como CPU al microprocesador 8085.

PROGRAMACION Y DESARROLLO DE MICROCOMPUTADORES BASADOS EN MICROPROCESADORES

Hasta este punto se ha abordado el estudio del microcomputador desde el punto de vista del hardware sobre todo. Pero un microcomputador además de la definición del conjunto de dispositivos e interconexiones que lo configuran precisa del software de base, tanto a nivel de sistema operativo como de programas de ayuda y desarrollo para poder funcionar.

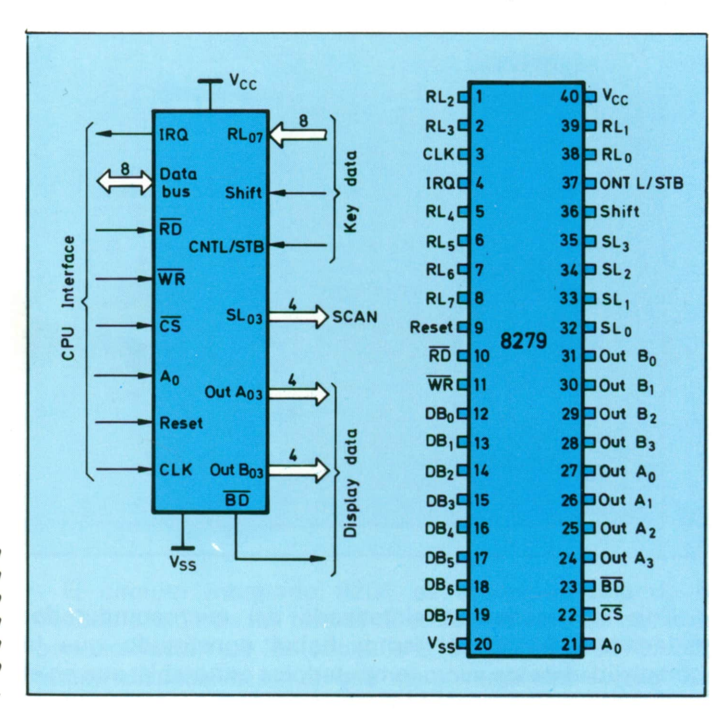


Figura 19. A la izquierda se representa el diagrama de bloques del circuito integrado 8279, y a la derecha se especifica la función de cada pin o patilla.

Cuando la capacidad de integración de los fabricantes de dispositivos microelectrónicos no alcanzaba la sofisticación actual, el proyectista que se enfrentaba a un problema procedía, generalmente, a desarrollar un programa de aplicación partiendo del hardware disponible, sin ocuparse

de profundizar en él y teniendo como único objetivo la aplicación del sistema para la resolución del problema.

Este planteamiento sigue siendo válido en el caso de que la unidad central del microcomputador sea un microprocesador, pero la versatilidad de éstos añade nuevas posibilidades.



Los periféricos de computador pueden adoptar una gran variedad de aspectos funcionales, pero este PROS 80 A es portátil, lo que permite su traslado a cualquier lugar. (Cortesía: Novatronic).

Cuando se pretende llevar a cabo el diseño de un microcomputador que deba intercambiar información con el exterior es preciso, con anterioridad a la realización del programa de aplicación de que se trate, tener definido de una forma completa el hardware: microprocesador para la CPU, las memorias, los circuitos de entrada/salida, etc.

Pero ante la necesidad de interconectar periféricos, los microprocesadores, como unidad central de los microcomputadores, añaden una nueva problemática. Un mismo periférico puede ser atendido de muy diversas formas, y cuanto más rápidamente se pretenda intercambiar información con él más complejo resulta el hardware o el software del sistema.

Es por ello que, en este punto, el proyectista debe alcanzar una solución de compromiso entre complicar el hardware o simplificar el software, o a la inversa.

Antes de tomar una decisión frente a esta disyuntiva, es necesario conocer cuáles son las características del microcomputador que se ven influidas por un aumento de la complejidad del software.

En primer lugar, a medida que la complejidad del software aumenta, la capacidad de almacenamiento de la unidad de memoria debe aumentar también. En la práctica, la disponibilidad actual de circuitos integrados especializados en la memorización de información hace que este inconveniente sea de menor importancia. Pero el inconveniente es más grave en tanto en cuanto la mayor complejidad del software hace que el tiempo de ejecución de un determinado proceso se incremente. Este inconveniente es mayor en caso de que el microcomputador trabaje en tiempo real.

Se entiende que un computador trabaja en tiempo real cuando las operaciones que realiza se llevan a cabo durante el tiempo que dura el proceso físico en cuestión, de manera que los resultados de la información procesada por el computador pueden usarse para controlar el proceso físico.

A la vista de estos puntos resulta evidente que la solución para el proyectista debe ser la de reducir al máximo el hardware, siempre que el tiempo de ejecución de las operaciones esté dentro de límites tolerables. Esta es pues una solución de compromiso.

Sintetizando las fases de diseño de un microcomputador, éstas se pueden resumir en los puntos señalados en el diagrama de la figura 21.

Es evidente que en esta fase inicial de diseño es imprescindible un adecuado planteamiento del problema frente al que se enfrenta el proyectista. El planteamiento del problema será correcto a partir del conocimiento de las funciones que el microprocesador debe realizar. Sólo una correcta definición de la funcionalidad del microcomputador permite abordar las siguientes etapas de diseño con

suficientes garantías de éxito. Para que la definición funcional no pueda diluirse en el tiempo, es muy importante redactar en este punto un pliego de especificaciones y condicionamientos técnicos que el sistema debe respetar. Cualquier modificación debe ser estudiada con detenimiento, sobre todo su posible influencia sobre todas las demás y a la vez sobre el estado del proyecto en el momento en que dicha modificación se produce.

En la elección del microprocesador para que opere como CPU del microcomputador inciden diversidad de factores que ya han sido tratados abundantemente a lo largo de anteriores libros.

Los circuitos periféricos y su interconexión han sido también tratados en anteriores apartados, por lo que se van a reforzar principalmente en los próximos libros las consideraciones de diseño relativas al software de los microcomputadores.

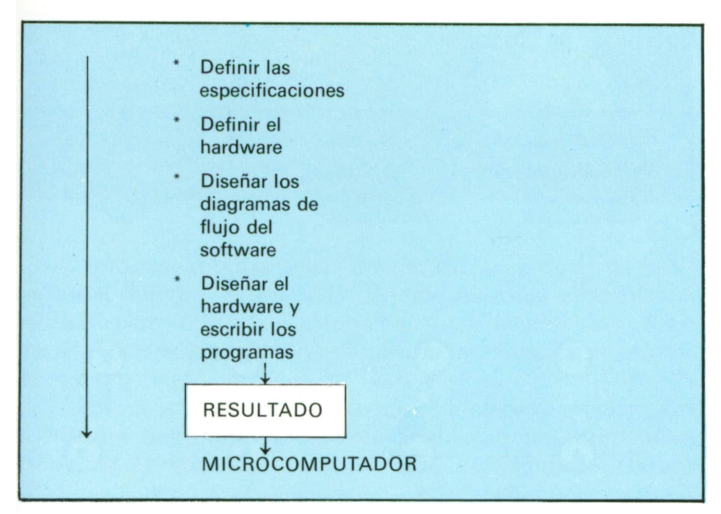


Figura 21. Fases de diseño y puesta en marcha de un microcomputador.

No resulta fácil dar «recetas» sencillas para configurar el hardware de un microcomputador. Como el lector podrá comprobar en los próximos libros, mucho más complejo resultará concretar y llevar a cabo el software necesario para que el microcomputador cumpla las especificaciones.

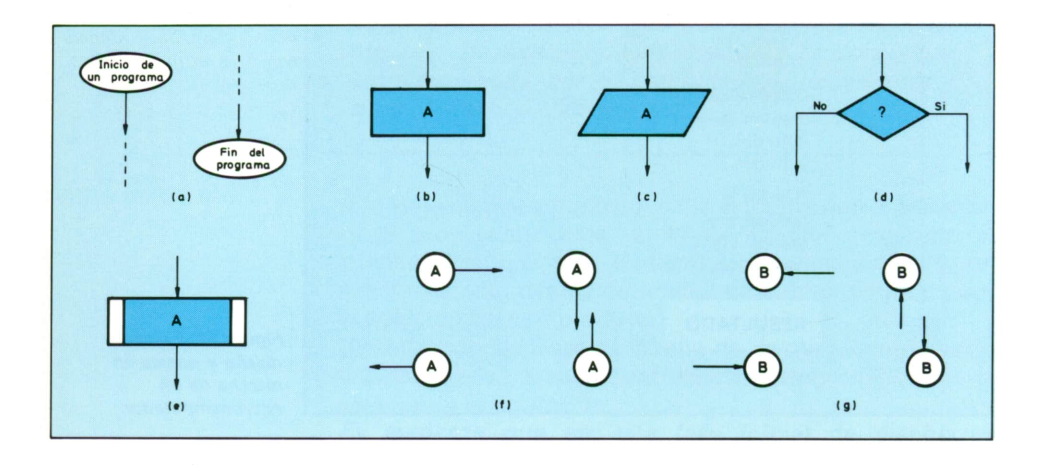
LOS DIAGRAMAS DE FLUJO DE UN PROGRAMA: SIMBOLOGIA NORMALIZADA

Una vez concretadas las especificaciones del sistema y decidido el hardware del microcomputador (de hecho ya se ha resuelto en este punto el compromiso hardware-software) se conoce ya el algoritmo del programa solución del problema y se puede pasar ya a su solución.

El algoritmo de un programa consiste en un conjunto de reglas o procedimientos determinados por los que se puede resolver un problema mediante un cierto número de pasos.

La representación esquemática mediante un gráfico del algoritmo, en donde se señalen las operaciones a llevar a

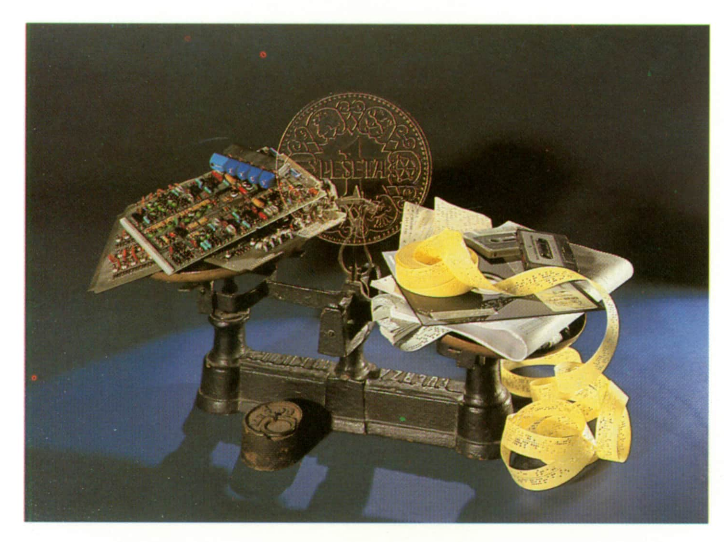
Figura 22. a) Símbolo de inicio y final de un programa; b) Ejecución de una operación o instrucción dentro del diagrama de flujo de un programa; c) Entrada y salida; d) Toma de decisión o de salto condicional con dos posibles salidas; e) Formato de salto a subrutina. Símbolo de conexión entrante (f) o saliente (g) del diagrama de flujo del programa



cabo y la interrelación entre ellas, constituye el punto de partida para configurar el programa que permita resolver el problema concreto que se tiene planteado.

La representación gráfica del algoritmo se denomina de

diversas formas, pero las más frecuentes son las de *diagrama de flujo*, o bien *ordinograma*. Con el fin de facilitar la confección de diagramas de flujo y homogeneizar de esta forma los diversos estudios realizados, existen unos símbolos normalizados que representan cada una de las acciones que debe llevar a cabo el computador.



El software cada vez «pesa más» con respecto al coste económico comparativo que supone el hardware. La tendencia es a que las máquinas se abaraten, y se eleve el coste de los programas.

— Símbolos de comienzo o final de un programa o de parte del mismo (figura 22*a*). Puede utilizarse este mismo símbolo para indicar un retorno al final de una subrutina. Notar la diferencia en la incidencia de la flecha en los casos de inicio o final.

— Símbolos de operación o acción durante el proceso. En el interior del símbolo, como en el caso anterior, suele escribirse el tipo de operación que corresponda (figura 22*b*).

— Símbolos de entrada y salida: son similares a los de operación, pero con inclinación del lado superior a la derecha, según se muestra en la figura 22*c*.

— Símbolo de decisión es el que se usa cuando debe de tomarse una decisión en torno a un resultado. Los dos terminales de salida representan las dos posibles alternativas básicas «SI» o «NO», aunque no son las únicas, ya que

determinadas tomas de decisión pueden permitir más de dos alternativas (figura 22*d*).

— Símbolo de llamada a subrutina: suele utilizarse el símbolo que se representa en la figura 22*e*.

Cuando un programa es demasiado largo para que quepa en un único documento, o bien en el caso de que sea preciso efectuar saltos condicionales se han creado símbolos de conexión normalizados como los que se muestran en la figura 22*f* y *g*.

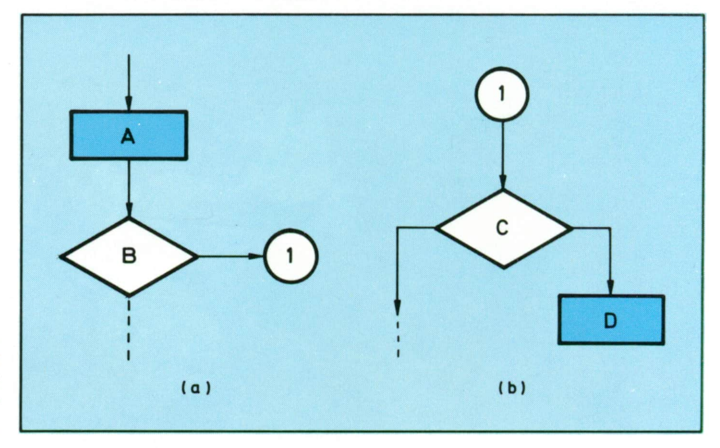


Figura 24. Ejemplos de aplicación de los símbolos de conexión: a) Conexión de salida; b) Conexión de entrada.

RACIONALIZACION DE LA PROGRAMACION: SUS TECNICAS

Para representar el diagrama de flujo de un determinado algoritmo no existen procedimientos ni reglas preconcebidas, las técnicas de confección de diagramas de flujo y de programación se perfeccionan de forma considerable a través de la práctica y, hasta hace pocos años, programar consistía en confeccionar una determinada secuencia de instrucciones a partir del adecuado conocimiento de un lenguaje, con el fin de realizar la función deseada sin atenerse a procedimientos establecidos. Con ello se conseguía que un programa, una vez escrito, fuera de difícil comprensión para otros especialistas, ya que la falta de una teoría homogénea diversificaba enormemente las técnicas de programación.

Si bien esta situación no ha sido erradicada totalmente, la *programación estructurada* ha venido a uniformizar grandemente las técnicas de programación, permitiendo a partir de fundamentos generales una mayor homogeneidad en la estructura de los programas.

La programación estructurada consiste en definir un conjunto de estructuras fundamentales por un lado y procedimientos de combinación de estas estructuras por otro con el fin de obtener los programas.

En primer lugar se van a tratar las estructuras fundamentales y posteriormente su combinación.



Microcomputador modular para gestión, teclado, CPU y discos flexibles. Obsérvese que el diseño va dirigido para que no desentone con los ambientes de la empresa u hogar en donde se ubicará. (Cortesía: Burroughs).

Las estructuras fundamentales más comúnmente usadas son:

— Secuencial.
— Decisión.
— Decisión reiterativa.
— Subrutina.

La *estructura secuencial* corresponde a la realización, una tras otra, de determinadas operaciones, sin que haya otro tipo de acción o instrucción distinta. Su representación gráfica se muestra en la figura 26 donde, como se recordará, cada uno de los rectángulos indica la ejecución de determinada operación.

La *estructura de decisión* será, en forma general, según se muestra en la figura 27. Si existe más de una operación como resultado de la decisión la estructura se representará según aparece en la figura 28. La figura 29 muestra una decisión de tipo particular.

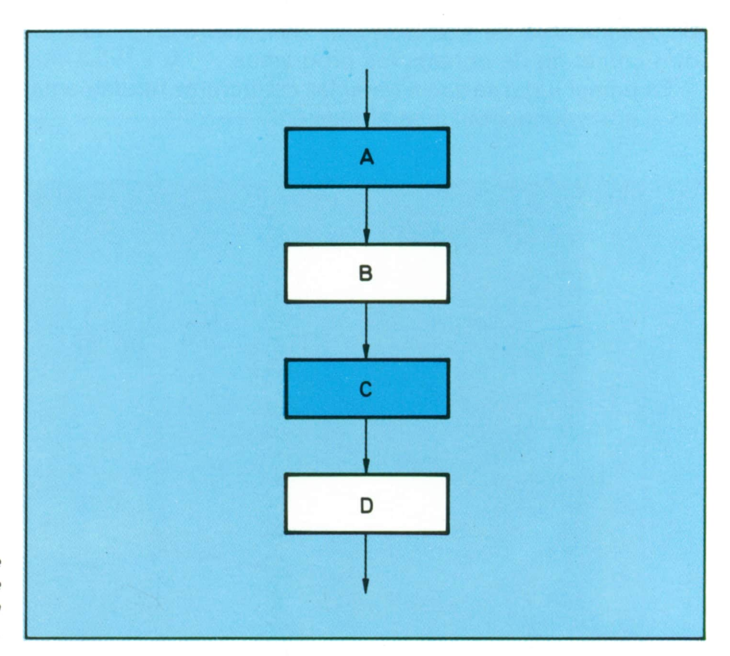


Figura 26. Diagrama de flujo de una secuencia de instrucciones en un programa.

En la figura 30 se muestra el caso de varias posibilidades de decisión, en el supuesto de que el microcomputador sólo disponga de dos alternativas como resultado de la decisión.

La *estructura de decisión reiterativa* aparece a partir de la necesidad de repetir una parte del proceso previsto en el programa, necesidad que por otra parte se presenta con relativa frecuencia. Este problema puede resolverse de muy diversas formas, y está claro que la más simple consiste e concatenar la misma operación tantas veces como sea preciso (figura 31).

Si el número de veces que debe repetirse determinada

operación es muy grande, se consumirá una buena porción de memoria de una forma verdaderamente poco eficaz. Es por este motivo que se prefiere una solución del tipo

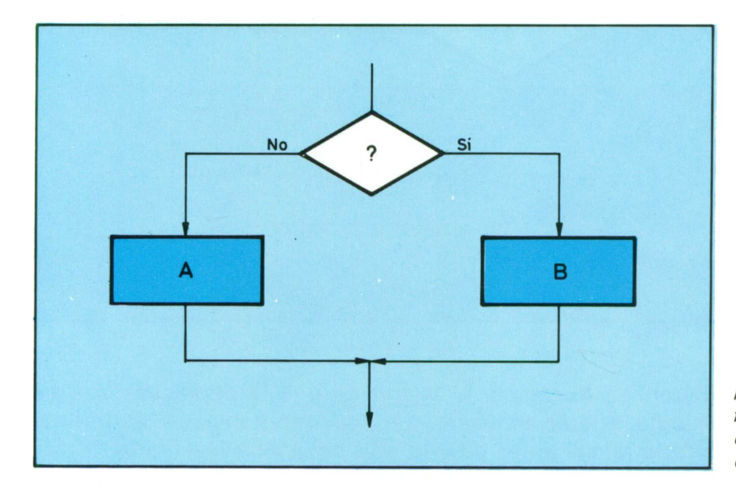


Figura 27. Diagrama de flujo de una instrucción de toma de decisión con dos salidas.

mostrado en la figura 32*a*, en donde se aprecia que el programa estará ejecutando de forma continuada la operación *A* en tanto no se modifique la condición oportuna que le permita seguir adelante. A una estructura de este tipo se le

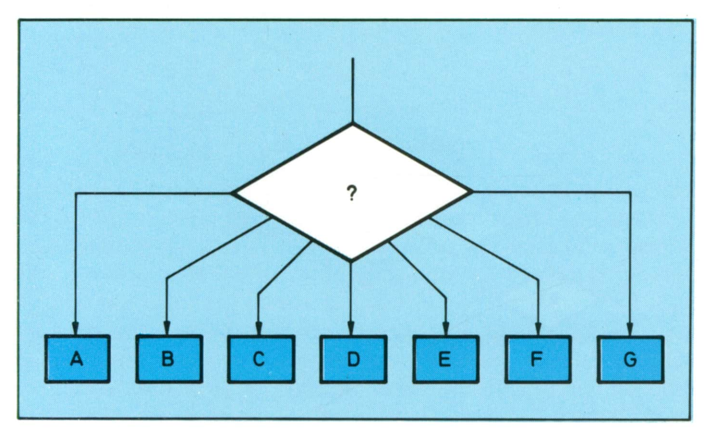


Figura 28. Símbolo de una instrucción de toma de decisión con varias salidas.

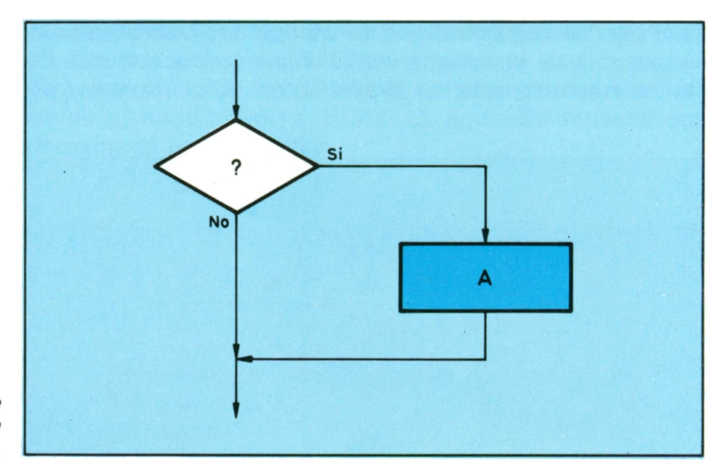


Figura 29. Símbolo de toma de decisión particular o única.

denomina *bucle* por el hecho de que la operación que se ejecuta sucesivas veces sitúa al programa en el mismo estado inicial antes de que ésta haya sido ejecutada.

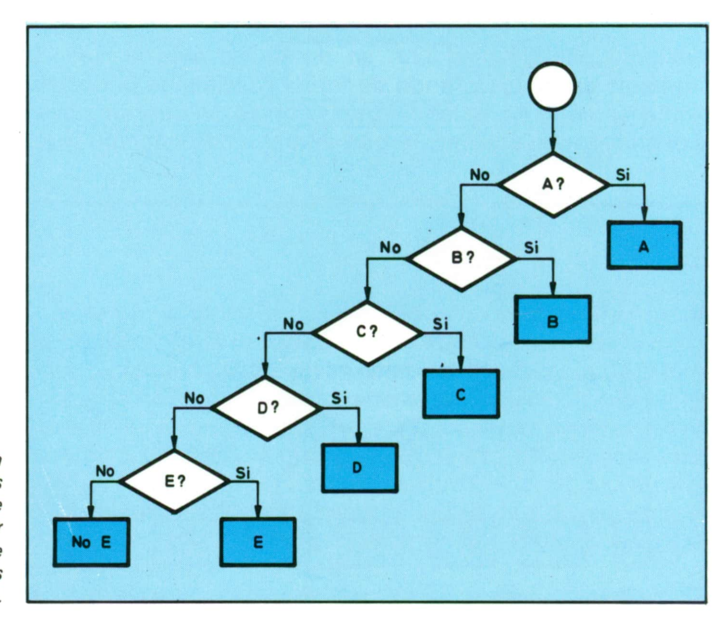


Figura 30. Caso de toma de decisión con múltiples alternativas, en el caso de que el microcomputador sólo disponga de toma de decisión de dos alternativas.

Cuando se pretende que un determinado proceso sea ejecutado un número concreto de veces la estructura que se adopta es la mostrada en la figura 32*b*, en donde puede apreciarse que se presenta un determinado subíndice $A=N$ (siendo N el número de veces que se pretende se repita el

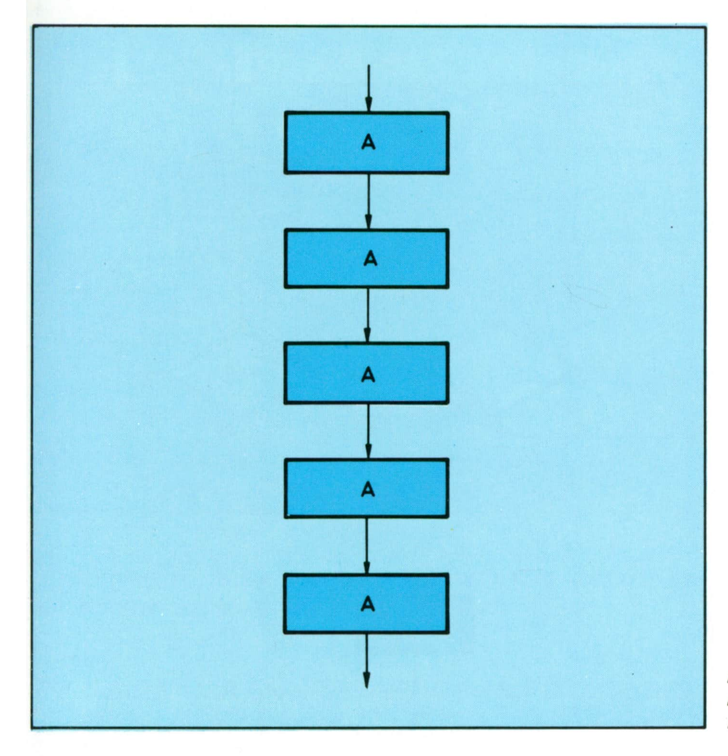


Figura 31. Repetición reiterativa de una instrucción.

proceso en cuestión). La decisión de finalizar la ejecución del proceso se producirá cuando el subíndice A sea cero. Una variante de la estructura anterior se presenta en la figura 32*c*, en la que el lector puede apreciar que tiene el mismo significado que la anterior.

La *estructura de subrutina* aparece cuando un determinado número de instrucciones se obtienen separadas del programa principal y, cuando estas instrucciones se han ejecutado, se regresa al programa principal mediante la

adecuada instrucción de RETORNO. La figura 33 muestra la estructura general de una subrutina y cómo está concatenada esta estructura dentro del diagrama de flujo del programa principal.

Aunque las estructuras mostradas no son las únicas a las que se les podría adjudicar rango de «normalizadas», no es menos cierto que estas estructuras pueden adoptarse con

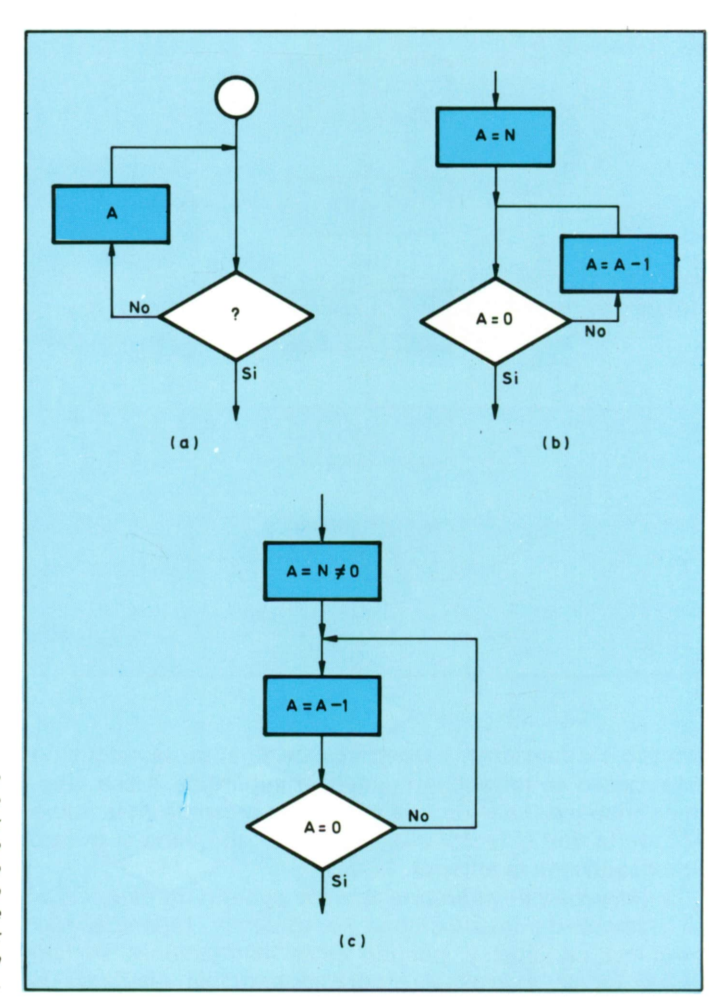


Figura 32. a) Diagrama de flujo de un bucle condicionado; b) Diagrama de flujo de un bucle condicionado con limitación del número de repeticiones mediante indicativo numérico; c) Variante de la configuración anterior.

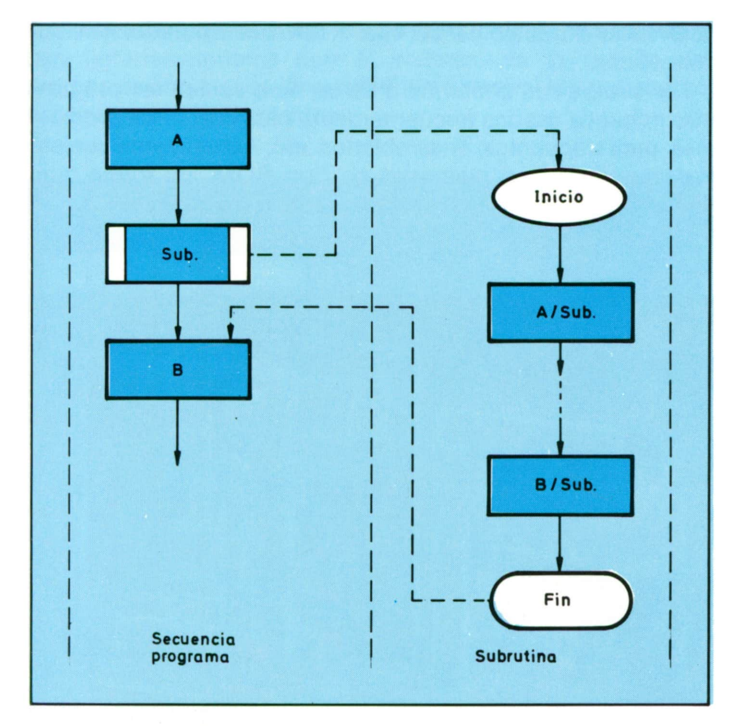


Figura 33. Fragmento del diagrama de flujo de un programa, incluyendo el salto a una subrutina y la vuelta a la secuencia del programa principal.

prácticamente todos los lenguajes y todas las posibles estructuras hardware de los microcomputadores.

EL PROGRAMA EN LOS MICROCOMPUTADORES

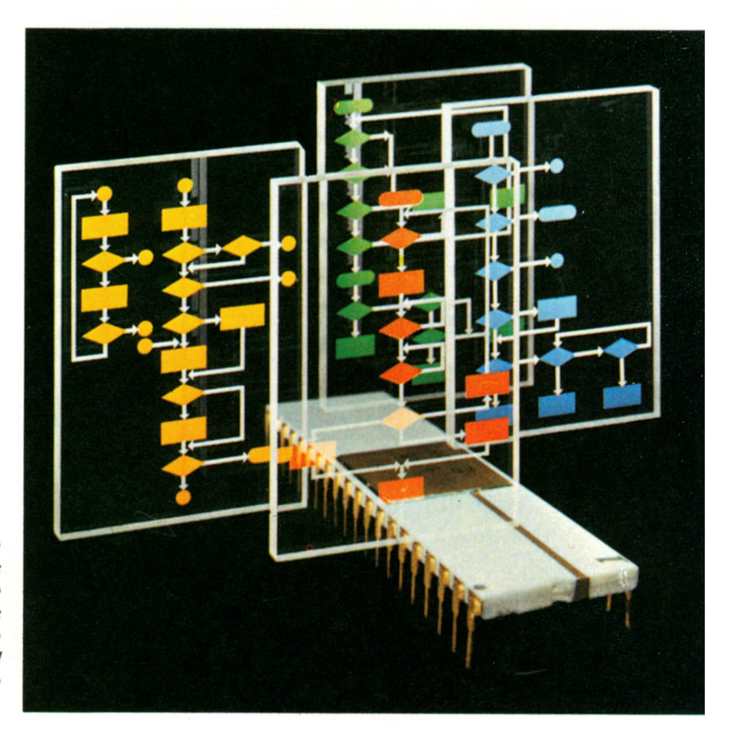
Es conocida por el lector la dificultad de realizar programas en lenguaje máquina, motivo por el cual los lenguajes de alto nivel tienen la importancia que han alcanzado en el desarrollo y proliferación de los microcomputadores. Los programas de los microcomputadores pueden enmarcarse dentro de tres categorías:

— Programas *utilitarios* o de *servicio.*
— Programas de *explotación.*
— Programas de *aplicación.*

Cuando se pone en marcha un computador es preciso proporcionarle determinados programas para que pueda

desarrollar el trabajo para el cual el microcomputador ha sido concebido.

Los programas *utilitarios* o de *servicio* suelen ser programas mínimos usados frecuentemente para introducir programas, para traducirlos, ensamblarlos, etc. Actualmente suelen ir desarrollados en memorias de tipo ROM, de forma que dichas instrucciones están disponibles a partir del momento de la conexión del sistema.



«Entrar» el programa en un microprocesador no es precisamente lo que se muestra en la figura. Los bits se transfieren eléctricamente cuando el microcomputador está en funcionamiento.

Los programas de *explotación* o *sistema operativo* del microcomputador, dotan a éste de la capacidad de dirigir todas las operaciones que llevará a cabo mediante los programas de aplicación. Constituyen la verdadera capacidad intelectual del sistema.

Los programas de *aplicación* ofrecen la solución al problema concreto que tiene planteado el programador y

son prácticamente el único medio de que dispone el usuario para interrelacionarse con el sistema. Es la familia de programas con mayor diversidad, en función de las distintas necesidades que se plantean.



Cada día es más conveniente que los microcomputadores e impresoras asociadas, sean capaces de procesar todos los colores; de este modo se facilita la identificación en pantalla y luego en el material impreso.
(Cortesía: Tektronix).

Cada uno de los programas, sean de la familia que sean, están escritos en determinado lenguaje de programación, encargándose el mismo microcomputador de traducirlos a nivel de «ceros» y «unos» lógicos, único «lenguaje» que entienden los computadores.

LENGUAJE MAQUINA

El conjunto de instrucciones capaces de ser comprendidas por un microprocesador, CPU de un microcomputador, constituye el lenguaje máquina de éste. Tales instrucciones deberán estar escritas mediante series de «1» y «0» lógicos, esto es: en el sistema binario de numeración. Los lenguajes máquina están prácticamente en desuso, pues la casi totalidad de los microprocesadores soportan lenguajes más complejos o de alto nivel. No obstante, para programas muy

cortos resulta recomendable a veces acoplarlos a este lenguaje, con lo que se gana velocidad en la ejecución de las instrucciones, ya que este lenguaje optimiza la operatoria del microprocesador.



Computador de Secoinsa, serie 20, para ofimática y gestión de datos en general.

Los motivos por los que los lenguajes máquina no se están utilizando apenas son diversos:

— Los códigos de operación son muy difíciles de recordar al no existir ninguna relación entre la denominación de la instrucción y su función: todas ellas están constituidas por «0» y «1».

— Las direcciones de los datos a manejar son igualmente difíciles de recordar.

— Al cargar los programas, teniendo en cuenta que deben ser introducidos tanto datos como instrucciones en código binario, es muy sencillo cometer errores y éstos son difíciles de detectar.

— Los programas escritos en lenguaje máquina son los que presentan un mayor grado de incompatibilidad entre distintos sistemas, ya que la codificación de cada instrucción es biunívoca para cada tipo de microprocesador.

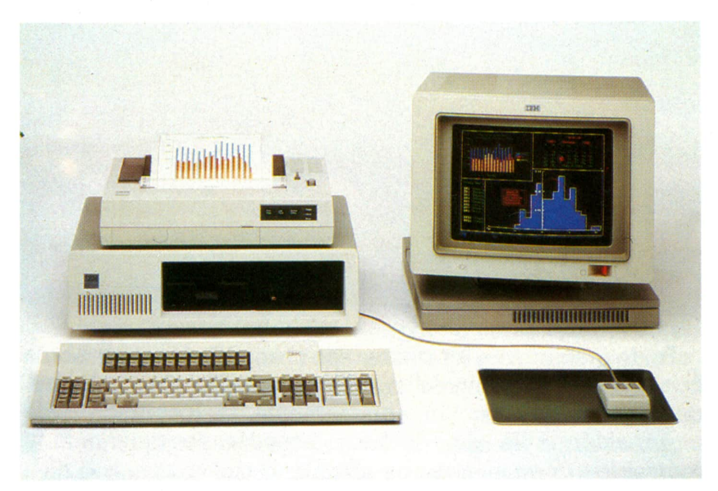
— Dada la densidad de los programas escritos en este lenguaje, ni el propio programador que lo ha realizado es

capaz, con frecuencia, de entenderlo al cabo de un cierto tiempo de haberlo llevado a cabo.

Por todo lo expuesto se comprende que el lenguaje máquina sea muy poco usado a pesar de su innegable importancia didáctica y formativa, puesto que con la expresión formal de las instrucciones en codificación binaria se puede comprender mejor el funcionamiento intrínseco de los microprocesadores.

LENGUAJE ENSAMBLADOR

Puesto que el único sistema de numeración y de codificación que es capaz de entender un microprocesador es el binario, sea cual sea el lenguaje que se utilice para realizar programas, deberá efectuarse sobre dicho programa algún tipo de manipulación para que su contenido sea inteligible para el microprocesador.



Cuando un microcomputador trabaja con varios periféricos, hay que cuidar en gran medida la tecnología seguida en la interconexión, para evitar riesgos en la transferencia de datos. (Cortesía: IBM).

Llegados a este punto es conveniente aclarar el doble significado que en el argot informático se le da a la palabra «ensamblador».

Por un lado, *ensamblador* es el nombre de un lenguaje simbólico que sirve para realizar programas y ser ejecutados

por los microprocesadores. Por otro lado, se denomina ensamblador a un determinado programa de traducción (o ensamblaje, de ahí su nombre) que se encarga de convertir en lenguaje máquina un determinado programa que ha sido escrito un *lenguaje simbólico* o *mnemónico.* Volviendo al lenguaje ensamblador, conviene señalar que éste permite utilizar símbolos y denominaciones para designar instrucciones, datos y direcciones, de manera que la uniformidad de la codificación binaria da paso a un formalismo mucho más rico y, sobre todo, más potente para indicar el significado de determinadas instrucciones u operaciones.



Aplicación de un microcomputador destinado como herramienta de trabajo para científicos y técnicos. En este caso, se introducen y procesan los datos de análisis químicos y bacteriológicos de suelos, aguas y aire. (Cortesía: Hewlett Packard).

Es de gran interés en el lenguaje ensamblador el hecho de que una dirección pueda identificarse mediante un nombre, de manera que facilita enormemente la localización de información a la que interese acceder en determinado momento.

Un tercer factor de enorme interés que aporta el lenguaje ensamblador es el de que las constantes se pueden especificar en distintos códigos de numeración y sin necesidad de conversión previa. De esta forma, para la definición de las constantes pueden coexistir los códigos decimal, octal, hexadecimal, caracteres BCD, ASCII, etc.

Es conveniente resaltar que el ensamblador permite

escribir comentarios dentro del propio programa, cosa que favorece enormemente el seguimiento y posterior optimización de cualquier programa que haya sido escrito en este lenguaje.



Incluso las unidades de microcomputador pueden estar preparadas para trabajar con todo tipo de lenguajes, de modo que puedan ser compatibles con un computador central. (Cortesía: Texas Instruments).

El ensamblador tiene variantes de interés que, aunque a título enunciativo, conviene incluir antes de dar por finalizado este apartado:

— *El cross-ensamblador.* Este programa no se ejecuta en el microcomputador al cual va destinado, sino en otro sistema más potente respetando las reglas de ensamblaje del primero.
— *El macroensamblador.* Es una variante del ensamblador en el que cada instrucción da origen a varias operaciones más sencillas, de forma que no es preciso referirse a ellas individualmente sino que se ejecutan a partir de la macroinstrucción designada por el macroensamblador.
— *El trasladable o reubicable:* Es un ensamblador que trabaja sobre instrucciones dirigidas relativamente respecto a una referencia u origen que establece el programador. Como contraposición a este ensamblador

existe el ensamblador absoluto, que mantiene las instrucciones en direcciones fijas que previamente ha definido el programador.

LENGUAJES EVOLUCIONADOS DE ALTO NIVEL: FORTRAN, COBOL, BASIC, PL1, APL, PL-M, PL-M PLUS

En este apartado no se pretende realizar un estudio de estos lenguajes de programación, ni mucho menos evidenciar su capacidad y potencia para ordenar la ejecución de instrucciones. Más bien se pretende enunciar cuáles son los lenguajes de programación más comúnmente utilizados, señalando el campo de aplicación más frecuente en cada caso.

El lenguaje FORTRAN, cuya denominación proviene de su significado en inglés *fórmula translation*, tiene su principal ámbito de aplicación en el terreno del cálculo científico y técnico. Son muchos los sistemas de mediana y gran potencia que lo incorporan, si bien su uso está ligeramente en declive ante la mayor potencia y mejor especialidad del lenguaje ALGOL.

El COBOL (*common business oriented language*) es un lenguaje especializado en la problemática de la gestión de los negocios. Es utilizado por todo tipo de computadores y su preponderancia como lenguaje de gestión es hoy día indiscutible.

El lenguaje de programación BASIC (*beginnerus all-purpose symbolic instruction code*) es el más universal de los lenguajes actuales. La gran ventaja que presenta es su facilidad para ser aprendido y la cantidad de equipos que lo utilizan. Aunque no está especializado en ningún ámbito en concreto satisface la mayoría de necesidades que se pueden plantear en la actualidad, incluso para ser utilizado por programadores experimentados, los cuales mostraron en principio una cierta reticencia a su uso argumentando que el motivo de su creación fue el permitir el acceso a la programación a los no expertos.

Con el paso del tiempo, el BASIC se ha ido convirtiendo en un lenguaje muy completo del que se han escrito gran cantidad de versiones distintas, si bien todas ellas enormemente próximas.

El PL1 (*programming language one*) fue un lenguaje que alcanzó gran proliferación con la serie 360 de IBM. Es un lenguaje de carácter universal. El APL, cuya denominación proviene de *a programming language*, es un lenguaje con una perspectiva de futuro notable, si bien el número de detractores es también considerable. Su utilización es prácticamente universal desde el punto de vista de los ámbitos a los que se suele aplicar.

```
100 NEXT I
110 FOR I=1 TO V
120 A=INT (RND*N+1)
130 C (A)=C (A)+1
140 NEXT I
150 REM ORDENACION DE LA LISTA
170 FOR I=1 TO N-1
180 IF C (J)>=C (J+1) THEN 220
190 X=C (J):X $=C $ (J)
200 C (J)=C (J+1):C $ (J)=C$ (J+1)
```

Fragmento de un programa escrito en lenguaje BASIC.

El PL/M es una derivación del PL1, que se generó para ser aplicado en los computadores que usaran el microprocesador 8008 de Intel. Posteriormente ha sido adaptado para el microprocesador 8080. No es de los lenguajes más comúnmente usados en la actualidad, a pesar de que en la fecha de su lanzamiento su futuro aparecía prometedor.

El PL-M. PLUS fue desarrollado por National Semiconductors para sus sistemas de 16 bits IMP-16 y PACE.

Es evidente que no todos los lenguajes han sido enunciados aquí. Algunos como el ALGOL y el PASCAL, de más reciente aparición, presentan un futuro tremendamente prometedor, si bien es necesario hacer notar que estas expectativas se ven frecuentemente truncadas.

El programa editor: diseñando el programa

Como se ha explicado anteriormente, una vez se ha estructurado el diagrama de flujo de un determinado programa, que ha de ser la solución del problema que al programador se le presenta, pasa a escribirse el programa en el lenguaje seleccionado comprensible para el sistema

hardware de que se disponga, esto es, para el microcomputador.

Una vez se ha escrito el programa generalmente se graba en un soporte magnético o en una memoria externa, de forma que dicho programa pueda ser leído por el traductor mediante la utilización del periférico adecuado.

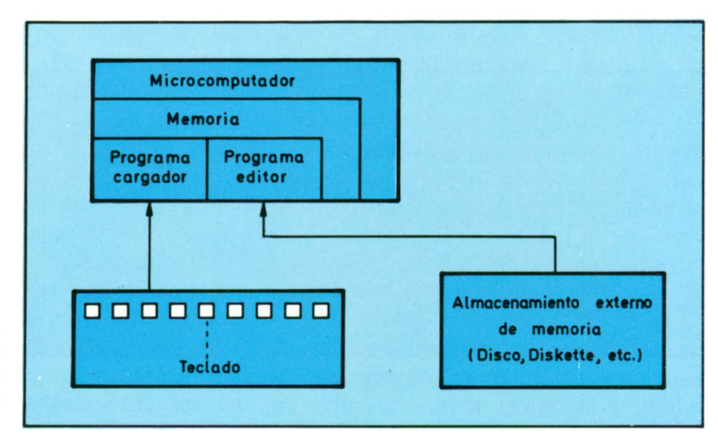


Figura 41. Configuración hardware mínima necesaria para introducir un programa en un microcomputador.

Las antiguas tarjetas perforadas o cintas magnéticas son sustituidas en los microcomputadores actuales por discos blandos (floppy disk) que actúan de soporte de dicha información. A partir del papel, el programador introduce los datos a través del terminal teclado, carácter a carácter, hasta que todo el programa completo ha sido introducido. El lector habrá adivinado que en este proceso de introducción del programa en el microcomputador no es difícil cometer algún tipo de error. Es fácil imaginar que se debe disponer de algún tipo de ayuda para evitar que cada puesta en marcha de un programa se convierta en una tarea difícil, mucho más difícil de lo que ya resulta en la realidad. En la práctica, los modernos sistemas de soporte de la infomación, sean semiconductores (memorias en circuito integrado), magnéticos (floppy disk) u otros, permiten el acceso aleatorio a la información de manera que resulta bastante sencillo, una vez detectado, corregir cualquier tipo de error que pudiera producirse.

Esquemáticamente, la situación que se presenta al introducir un programa en un microcomputador se muestra en la figura 41.

Como el lector habrá apreciado, en la figura aparece un programa cuya misión es introducir en el computador las órdenes que va dictando el programador a través del teclado. Dicho programa, el *cargador*, está grabado en una memoria pasiva del computador y se pone en marcha automáticamente, tras pulsar el conmutador correspondiente, posicionando la primera dirección de memoria en la que va a intoducirse la instrucción en el contador ordinal de programa.



La redacción de periódicos, agencias de noticias, centros públicos, etc., exige disponer de microcomputadores para agilizar la elaboración y compaginación de textos y datos estadísticos. (Cortesía: Diario «La Vanguardia»).

Este programa cargador, si además de cargar el editor sirve para poder observar el contenido de la memoria y modificarlo si conviene, se pasa a denominar *programa monitor.*

El *editor* es un programa auxiliar de programación que sirve para corregir y editar los programas fuente generados por el programador en el mismo lenguaje que el programador ha utilizado para llevar a cabo el programa.

En la figura 43 se puede apreciar el diagrama de flujo que se debe llevar a cabo para trabajar en programa editor,

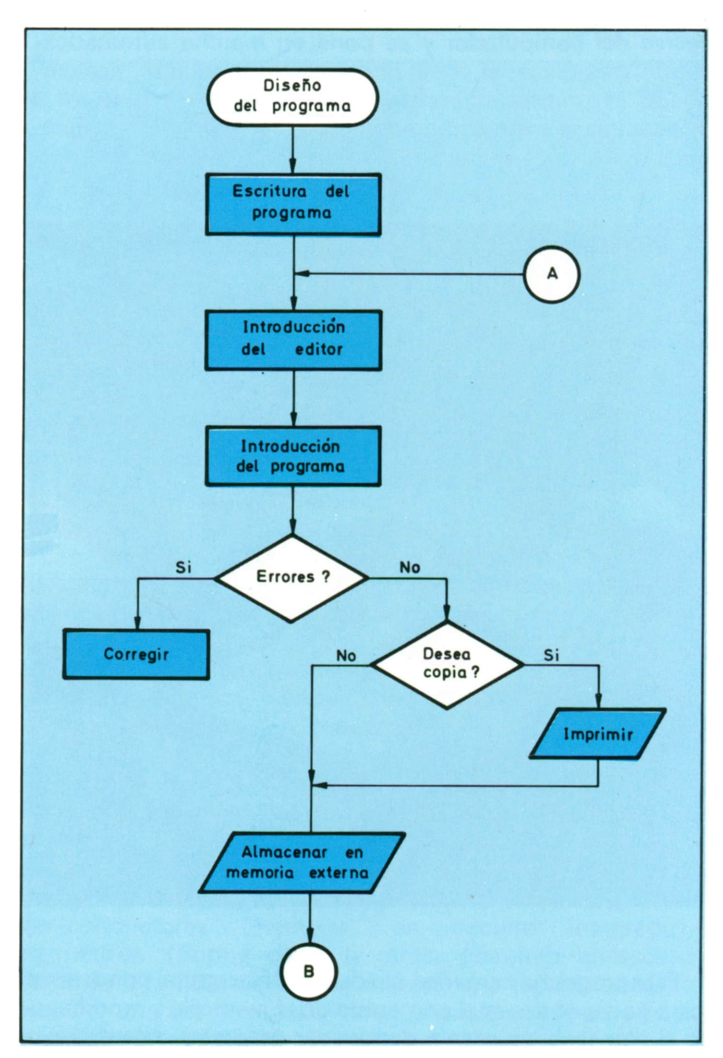


Figura 43. Diagrama de flujo de utilización del programa editor.

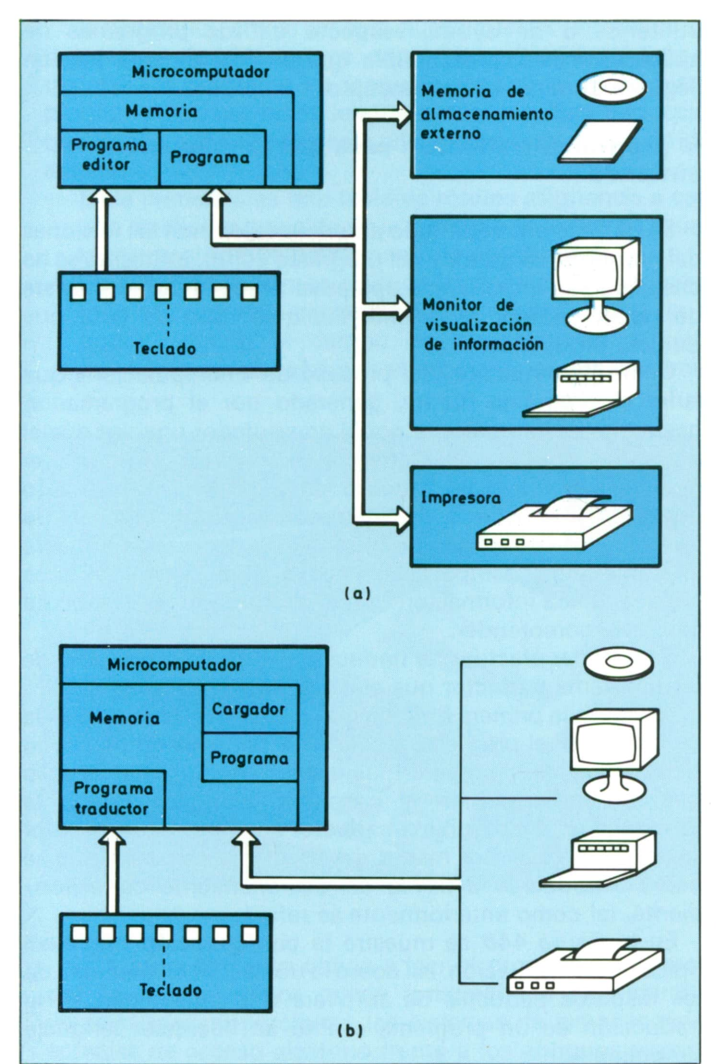


Figura 44. a) Configuración hardware más conveniente para trabajar con el editor; b) Configuración hardware más conveniente para trabajar con el programa traductor.

siendo la estructura hardware más conveniente la que se muestra en la figura 44*a*.

Es aconsejable que a esta altura el lector vaya comprendiendo la diferencia funcional y operativa de los programas

utilitarios o de ayuda, respecto de los programas de aplicación. Es imprescindible que al final de esta lección llegue a dominar estos conceptos.

El programa traductor: para que la máquina entienda

Se ha visto en el apartado anterior cuáles eran las misiones del programa cargador y del programa editor, asimismo se ha destacado su importancia operativa desde el punto de vista de poder corregir fácilmente cualquier tipo de error que pudiera producirse.

Como continuación del proceso de «manipulación» que sufre el programa fuente, generado por el programador, hasta que es interpretado por el computador una vez que el programa está correctamente editado, es necesario someter dicho programa a un proceso de traducción. Ya se dijo anteriormente que toda instrucción procedente de la utilización de un lenguaje simbólico era necesario traducirla de forma que al computador accedieran los ceros y los unos lógicos, única información que el computador es realmente capaz de comprender.

Para poder efectuar tal traducción es preciso disponer de un programa traductor que se encargue de esta tarea.

Así pues, la primera función que procede en este caso es la de introducir el programa traductor a partir de un periférico (lector de cinta, disco, etc.), dentro del computador. Es claro que, al igual que en el caso del programa editor, la introducción del programa traductor en el microcomputador se llevará a cabo por medio del programa cargador que se habrá introducido a través del accionamiento correspondiente, tal como anteriormente se refirió.

En la figura 44*b* se muestra la configuración hardware típica de esta situación, así como la ubicación en memoria de los distintos paquetes de software que intervienen en la traducción de un programa escrito en cualquier lenguaje simbólico.

En aquellos casos en que se dispone de memoria magnética externa, el operador del computador únicamente debe ordenar que se inicie la traducción y el programa cargador ya se ocupa de introducir el programa traductor, poniéndose éste automáticamente en marcha para desarrollar su función.

Es necesario señalar en este punto cuál es la función del programa ensamblador que, como ya se dijo, se encarga de traducir un programa en lenguaje simbólico en el mismo programa pero escrito en lenguaje máquina. Esta traducción que lleva a cabo el programa ensamblador se realiza en dos etapas:

— En la primera crea una tabla de rótulos asignando a cada uno de ellos la posición de la memoria en la que se situará la instrucción que posteriormente corresponda.
— En el segundo tiempo traduce instrucción a instrucción todo el programa, consultando los rótulos con el fin de poder calcular el campo de dirección en aquellas *instrucciones de salto* que el programa pueda contener.



Familia completa de computador y periféricos de la firma Digital.

Este procedimiento obliga a leer el programa dos veces, por lo que el lector puede imaginar las considerables ventajas que proporcionan los sistemas de almacenamieto actuales de acceso aleatorio frente a los antiguos sistemas de tarjetas perforadas, cintas continuas, etc.

En la figura 46 se muestra el diagrama de flujo que debe seguirse para llevar a cabo la traducción de un programa mediante el programa traductor.

Hay una característica del programa traductor que facilita enormemente la puesta en marcha de un programa, esta

característica es la detección de los errores de sintaxis del programa. ¿Qué se quiere expresar por error sintáctico? Se entiende por error sintáctico el que ha producido el programador al introducir el programa, por ejemplo: la escritura equivocada de una instrucción que no es capaz de reconocer el computador. Cuando ya se ha acabado la función de la traducción, puede imprimirse opcionalmente el

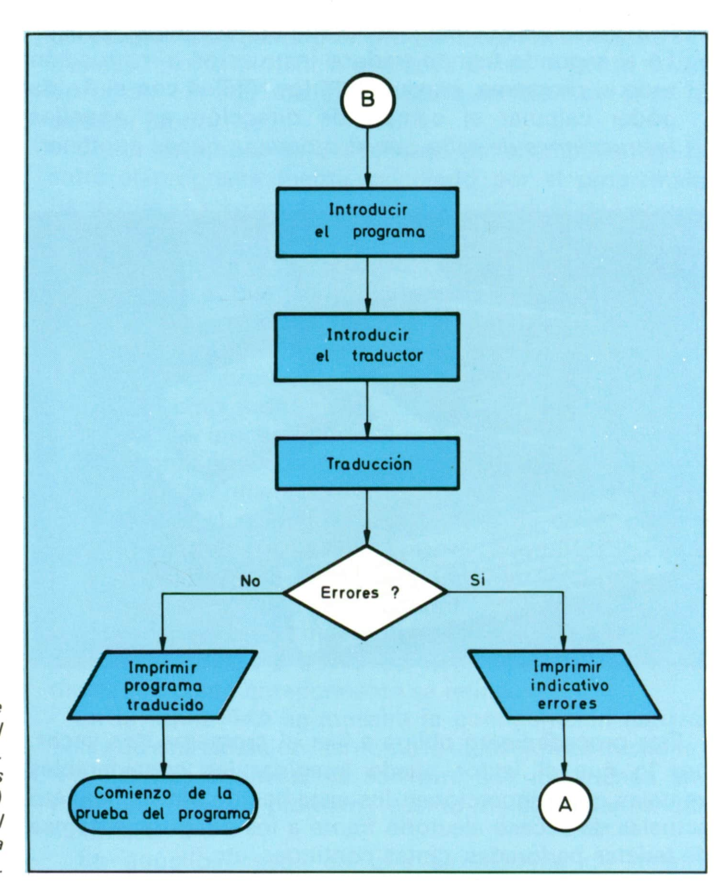


Figura 46. Diagrama de flujo de utilización del programa traductor. Nótese que los símbolos de conexión (A) y (B) están relacionados con el diagrama de flujo de la figura 43.

texto con unos indicativos de identificación del error o errores que en la introducción del programa hayan podido producirse.

Los errores detectados deberán corregirse mediante el editor o monitor y, una vez el programa sea conforme, se darán las instrucciones oportunas para que sea trasladado a la memoria externa, generalmente en lenguaje máquina.

El programa corrector: puesta en marcha y mejora

El resultado de la traducción permite disponer del programa realizado por el programador pero en lenguaje máquina, es decir, expresado como combinación de ceros y unos lógicos, único formalismo que es capaz de comprender el microcomputador.

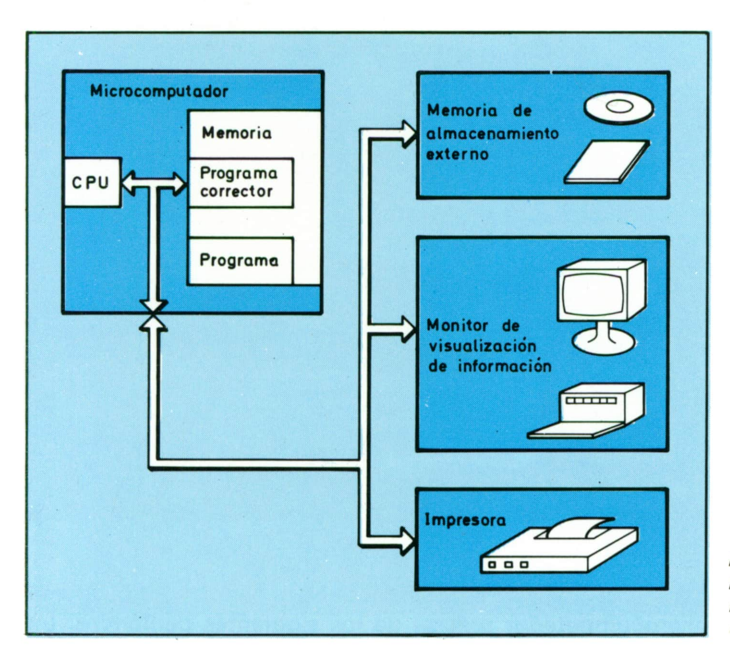


Figura 47. Estructura hardware mínima necesaria para trabajar con el programa corrector.

Una vez el programa ha sido traducido, es necesario probarlo con el fin de constatar si soluciona adecuadamente el problema que, mediante el mismo, se deseaba plantear.

La prueba o control de calidad de los programas y en general del software de los computadores, es cada vez más

un campo apasionante de investigación o trabajo y no existe, por el momento, una receta o procedimiento de actuación universal para certificar la correcta ejecución de determinado programa.

Para que un programa pueda probarse existe un programa auxiliar, llamado corrector (o *debugger* en inglés), el cual una vez cargado por medio del cargador se introduce en la memoria así como el lenguaje que se pretende corregir o probar. En la figura 47 se muestra la disposición que corresponde a esta situación.



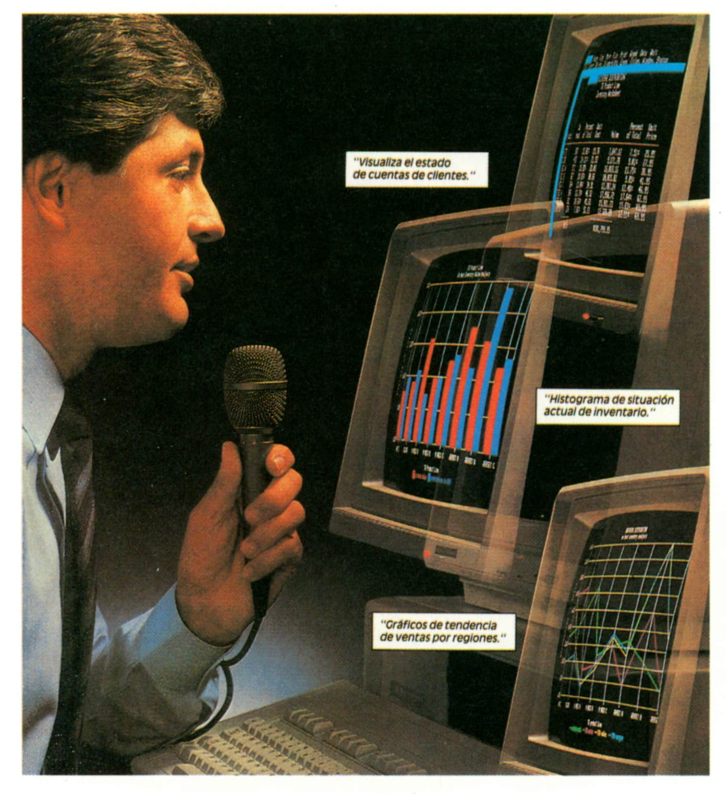
Los microcomputadores pueden servir para controlar la empresa desde el propio domicilio del director, ya que pueden estar conectados al computador central de la firma, y disponer así de todos los datos en cualquier momento del día o de la noche. (Cortesía: Epson).

Es evidente que, para poder llevar a cabo esta función, el microcomputador precisa de los siguientes periféricos: un teclado y una unidad de visualización, ya sea un monitor de TRC, display de LED de siete segmentos, de plasma, u otro similar. Es conveniente en estos casos disponer de una impresora que permita dejar constancia escrita del estado de cada uno de los pasos de evolución del programa.

Dejando a un lado la capacidad y potencia de los programas correctores que ofrecen los distintos fabricantes

de sistemas como software de soporte del equipo, pueden establecerse una serie de características comunes a la mayoría de programas correctores, que son las que se detallan a continuación:

— Mediante el programa corrector se puede modificar el programa que se está realizando aplicando las instrucciones adecuadas.



La gestión comercial queda mejorada si se logra disponer en la pantalla del computador del estado de cuentas de las empresas, la distribución de sus ventas, previsión de compras, etc., y mucho mejor si se realiza en forma de gráficos estadísticos. (Cortesía: Texas Instruments).

— Mediante el programa corrector se puede también iniciar la ejecución del programa que se prueba, de forma que sea posible hacerlo evolucionar instrucción a instrucción o bien pararse tras la realización o ejecución de determinados bloques de instrucciones por detenciones del mismo en los momentos apropiados.

El programa corrector permite además efectuar diferentes funciones entre la ejecución de dos instrucciones consecutivas, siendo ésta una de las características que diferencia a los distintos programas correctores de los diferentes sistemas.

Hay que señalar, con el fin de que el lector no se confunda, que mediante los programas correctores se puede probar el funcionamiento del programa, pero dicha prueba no es real, ya que el programa corrector no hace correr el programa en las condiciones reales de utilización como luego lo hará en la aplicación al problema planteado. Es por ello que el programa corrector, a pesar de la indudable ayuda que representa para el programador, tiene también importantes inconvenientes:

— La prueba del programa no se lleva a cabo en tiempo real, con lo cual las instrucciones no se ejecutan a la misma velocidad y de la misma forma que cuando dichas instrucciones son ejecutadas por el programa en aplicaciones reales, ésta suele ser causa de muchos problemas en bucles y subrutinas cuyo comportamiento es distinto en las condiciones de prueba que en las de aplicación real.

— Al no efectuar normalmente la prueba del programa con los periféricos y sistemas reales que posteriormente la soportarán, se presenta un grave inconveniente a pesar de la capacidad de simulación que proporciona el programa corrector, ya que los distintos protocolos que requieren cada uno de los distintos sistemas evidencian comportamientos no esperados en un determinado programa que ha sido probado con éxito. Este punto suele ser un verdadero quebradero de cabeza para los programadores experimentados, ya que añade una nueva problemática, la de la compatibilidad, a las graves dificultades de poner en marcha determinado programa.

La mayoría de los microcomputadores incorporan el programa corrector en la memoria ROM. Dado que a través del monitor de visualización suelen representarse las instrucciones para ser corregidas, es frecuente que dicho programa se denomine monitor.

La prueba y corrección del programa constituyen de hecho el último y previo paso a la utilización de un microcomputador.

Una vez especificado y configurado el hardware, dispo-

niendo de los programas auxiliares, del sistema operativo del cual se hablará más adelante, y del programa de aplicación que se deba utilizar, el usuario está en condiciones de «darle al interruptor» y ponerse a trabajar. A partir de este instante se enfrentará probablemente a nuevos problemas que poco o nada tienen que ver con el microcomputador que tiene entre sus manos.



El trabajo en las oficinas, puede ser mucho más agradable y menos ruidoso, utilizando microcomputadores en lugar de las viejas máquinas de escribir y las calculadoras manuales. (Cortesía: Philips).

CONÇEPTO DE SISTEMA OPERATIVO

Cuando el lector tenga un microcomputador ante sí lo primero que deseará es verlo funcionar. A través de lo que se le ha ido explicando hasta ahora habrá comprendido que, para que el microcomputador lleve a cabo una tarea determinada, es imprescindible que se introduzca un determinado programa en la memoria a través de alguno de

los periféricos que se han ido presentando y que tendrá oportunidad de estudiar con mayor detalle en próximos libros.

Dentro del microcomputador debe existir determinado programa que facilite el uso del computador de forma que pueda «conversar» con él a través de los periféricos de que disponga, que permita introducir los programas de aplicación y que, en fin, dote al microcomputador de la capacidad que le es propia.

Al conjunto de programas que permite sacar el máximo partido al microcomputador se le conoce con el nombre de sistema operativo (*operating system* en inglés). De hecho los programas cargador, monitor, traductor y corrector, que se han estudiado con anterioridad, forman parte de esta capacidad para ser utilizados que caracteriza a los microcomputadores. Pero no son éstos, claro está, los únicos programas que configuran el sistema operativo de los mismos.

A lo largo del tiempo se han seguido diversas tendencias a la hora de seleccionar un sistema operativo. Los fabricantes de hardware han seguido una tendencia cambiante, de forma que se ha pasado en más de una ocasión por las diversas posibilidades de utilizar un sistema operativo estándar o bien desarrollar un sistema operativo específico que, al menos a priori, permita optimizar el funcionamiento del hardware del microcomputador.

Evidentemente hay detractores y defensores de cada una de las dos opciones. Aquí no se va a defender ninguna de las dos posibilidades en detrimento de la otra, pero sí se va a señalar que la mejor ventaja de los sistemas operativos estándar es que, generalmente, están lo suficientemente experimentados como para no ofrecer duda alguna en cuanto a su correcto funcionamiento y, además, su elección permite frecuentemente aprovecharse de una amplia biblioteca de programas de aplicación general también muy experimentados habitualmente, de forma que se supla con estas ventajas la supuesta falta de especialización que para determinadas aplicaciones desee el usuario. Por el contrario, los sistemas operativos especialmente diseñados para un determinado hardware permiten a priori una optimización de las características del equipo, sobre todo para aplicaciones especializadas suelen ofrecer la mejor solución, la de más corta ejecución en el tiempo.

Aunque es obvio que en estas páginas no se van a tratar sistemas operativos particulares sí es conveniente, aunque sea brevemente, exponer cuáles son las características significativas de los sistemas operativos que son usados con más asiduidad en los diversos sistemas que actualmente se encuentran en el mercado: el sistema operativo CP/M y el sistema operativo OASIS. Es de destacar que existen otros sistemas operativos universales de reconocida reputación y es fácil suponer que algunos autores hubieran preferido glosar otros sistemas operativos de reconocida solvencia, pero hay que señalar que los dos referenciados son los de mayor proliferación entre los equipos comercializados en España.

El sistema operativo CP/M (S.O. para «Control Program for Microcomputers) fue desarrollado por la firma Digital Reseach en 1.974 y es sin duda el sistema operativo más extendido entre los microcomputadores actuales, a pesar de que su potencia no es más que discreta. Su gran ventaja consiste, evidentemente, en el gran número de equipos que lo incorporan, además de su amplísima biblioteca de programas de aplicación. Otra ventaja a tener en cuenta es que el sistema operativo CP/M es capaz de correr programas en la mayoría de lenguajes de alto nivel. El sistema operativo CP/M se adapta a cualquier equipo que esté basado en los microprocesadores 8080, 8086, Z-80 en sus versiones A y B,



Microcomputador destinado a cumplir funciones de ofimática, mejorando en gran medida la calidad del trabajo que efectúan los administrativos de las empresas que operan manualmente. (Cortesía: Epson).

en el Z-8000, etc. Es muy fácil de aprender y tremendamente simple y rápido a pesar de que, como ya se ha dicho, es poco potente.

Como contrapartida, el sistema operativo OASIS es uno de los sistemas operativos más potentes entre los que existen en la actualidad.

Debido a ello el sistema operativo OASIS ha sido adoptado por gran cantidad de fabricantes de sistemas.

Es interesante señalar que a pesar de su gran potencia, incluso equiparable a los sistemas operativos de muchos minicomputadores, el OASIS es muy sencillo de manejar y muy claro estructuralmente.

Sin profundizar más en este sistema operativo, es de interés subrayar que el OASIS es capaz de soportar siete lenguajes de programación distintos, encontrándose entre ellos los más universales y utilizados: Ensamblador, Basic, Cobol, Fortran, Pascal, C y Forth.

Como se señaló, no son éstos los dos únicos sistemas operativos existentes pero sí los más usados, siendo el OASIS el que mejores expectativas de futuro presenta en estos momentos.

Otros sistemas operativos también populares son: el MPM, el UNIX, el RT-II, el RTX-II, el DOS (disk operating system), etc.

Sistemas reales completos en un solo chip: microsistemas SAA 6000 y Z8.

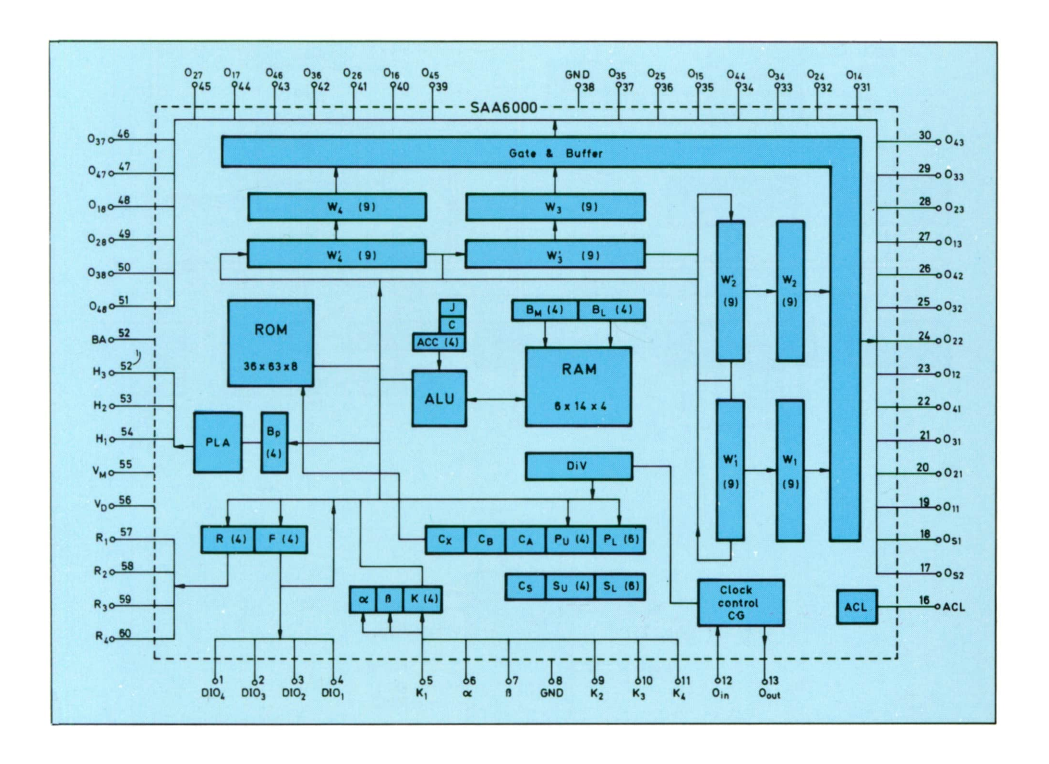
La tecnología microelectrónica permite una integración más y más completa de los circuitos electrónicos.

Un microsistema consiste en un circuito integrado de alta escala de integración en el que se encuentran incorporados la CPU, las memorias RAM, ROM (eventualmente EPROM), los circuitos de entradas y salidas, el reloj, etc. Habida cuenta de la compleja estructural de estos dispositivos, las posibilidades distintas que presentan son variadas, tanto o más en número de las familias distintas de microprocesadores que se ofrecen.

Lo mismo que en el caso de los microprocesadores se han tenido que elegir, entre la amplia oferta existente en el mercado, algunos microsistemas para que desde una óptica pedagógica permitan al lector familiarizarse o conocer al

menos la complejidad tecnológica y la funcionalidad de alguno de estos dispositivos.

Con este objetivo se han escogido el SAA 6000 y el Z8 por ser unos microsistemas utilizados abundantemente en los circuitos.



El SAA 6000 es un microsistema de ITT y fue iniciada su comercialización a inicios de 1.978. Está fabricado en tecnología CMOS y ha sido principalmente concebido y utilizado para productos portátiles sin fuente de alimentación conectable a la red, es decir: para equipos autónomos. Esto explica su reducido consumo (menos de 50 microamperios) alimentado a 3 V.

Figura 52. Diagrama de bloques del microsistema SAA 6000 de ITT.

El SAA 6000 contiene una memoria ROM de 2268 bytes de 8 bits cada uno, además de una RAM de 96 palabras de 4 bits, así como otros dispositivos funcionales, según se

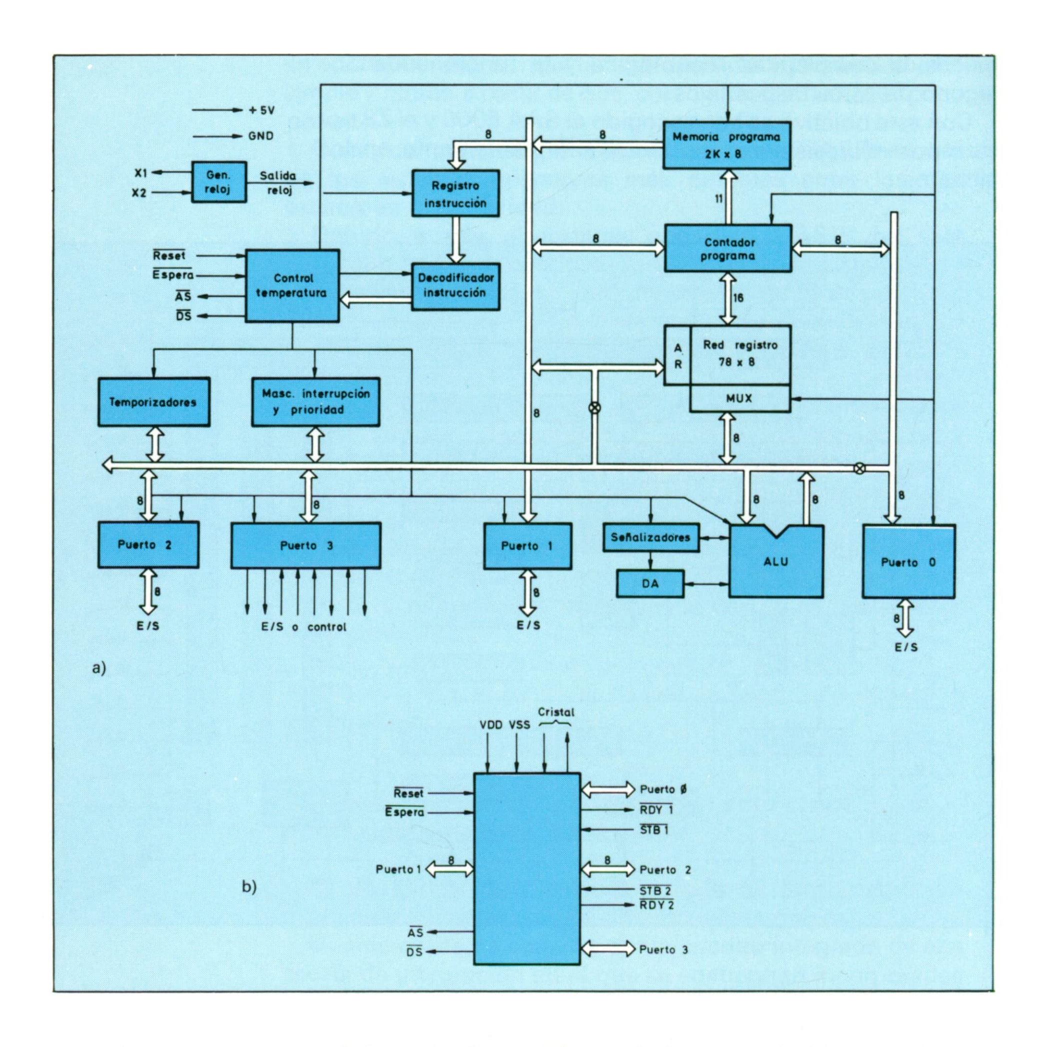


Figura 53. a) Diagrama de bloques del microsistema Z 8 de Zilog; b) Esquema de interconexión externa del Z 8 de Zilog.

señala en la figura 52 en el diagrama de bloques de este microsistema.

El Z8 constituye la versión monochip del popular Z80 de Zilog. Su diagrama general de bloques se muestra en la figura 53*a*, en la que puede apreciarse su estructura interna. En ella se observa una memoria ROM de 2 K octetos y una pila de 78 registros de 8 bits de uso general. La RAM que incorpora comprende 96 octetos.

